PROVABLE SECURITY SUPPORT FOR KERBEROS (AND BEYOND)

A Thesis
Presented to
The Academic Faculty

by

Virendra Kumar

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
College of Computing

Georgia Institute of Technology
August 2012

PROVABLE SECURITY SUPPORT FOR KERBEROS (AND BEYOND)

Approved by:

Professor Alexandra Boldyreva, Advisor
College of Computing
Georgia Institute of Technology

Professor Christopher J. Peikert
College of Computing
Georgia Institute of Technology

Professor Mustaque Ahamad
College of Computing
Georgia Institute of Technology

Professor Patrick Traynor
College of Computing
Georgia Institute of Technology

Dr. Vladimir Kolesnikov
MTS - Cryptography and Security Research
Bell Labs, Alcatel-Lucent

Date Approved: 27 April 2012

To my uncle,

late Dr. Kusheshwar Mandal,

who inspired and provided me

for whatever I am today.

ACKNOWLEDGEMENTS

My advisor Sasha Boldyreva has been a great inspiration, and I thank her for her guidance and support, and for being an excellent advisor.

I thank Mustaque Ahamad, Vladimir Kolesnikov, Chris Peikert, and Patrick Traynor for agreeing to be on my thesis committee, and Nate Chenette for proofreading parts of this thesis. I thank Yael Kalai for a brief research collaboration, and Chris Peikert for pointing out bugs in some of my research ideas. I also thank my co-authors Sasha Boldyreva, Vipul Goyal, Satya Lokam, and Mohammad Mahmoody for working with me.

I thank my seniors Vipul Goyal and Omkant Pandey, and my batchmates Abhishek Jain and Mayank Singh from my undergraduate institution. They have been very close friends over the years, and were the biggest driving force behind me pursuing Ph.D. studies.

During my Ph.D. studies I had multiple internship opportunities, and for that I thank Vipul Goyal and Satya Lokam from Microsoft Research, India; Zulfikar Ramzan and Sanjay Sawhney from Symantec Research Labs, Mountain View, CA; and Vladimir Kolesnikov from Alcatel-Lucent Bell Labs, Murray Hill, NJ.

Thanks to all my friends at Georgia Tech for making these past years pleasurable, they are just too many to list them all here.

I thank Shreyansh Chandra, Srinath Jagarlapudi, and Dushmanta Mohapatra for creating an enjoyable atmosphere at home for many years. They are the first friends I made in Atlanta.

I thank my family and my childhood friend Manish Ranjan, who are the dearest and closest to my heart, for their unconditional love and support.

Finally, I thank my wife Olga for making this life worth living, and for giving life and everything else to our (yet to be born) daughter.

TABLE OF CONTENTS

DEDICATION . . . . . **iii**

ACKNOWLEDGEMENTS . . . . . **iv**

LIST OF TABLES . . . . . **vii**

LIST OF FIGURES . . . . . **viii**

SUMMARY . . . . . **ix**

CHAPTERS

I INTRODUCTION **1**

1.1 Symmetric Encryption in Kerberos 2

1.1.1 Motivation 2

1.1.2 Related work 3

1.1.3 Our Contributions 5

1.2 Randomness Generation in Practical Protocols 7

1.2.1 Motivation 7

1.2.2 Related Work 9

1.2.3 Our Contributions 10

1.3 Revocation in Newer Types of Encryption 13

1.3.1 Motivation 13

1.3.2 Related work 15

1.3.3 Our Contributions 16

1.4 Organization and Conclusion 18

II PRELIMINARIES **20**

2.1 Notation 20

2.2 Symmetric Encryption Schemes and their Security 21

2.3 PRFs, MACs, and their Security 23

2.4 Encoding Schemes and their Security 25

2.5 Hash Function Families and their Security . . . 26
2.6 Pseudorandom Generator . . . 29
2.7 Bilinear Maps . . . 29

III SYMMETRIC ENCRYPTION IN KERBEROS . . . 31

3.1 General Profile . . . 31
3.1.1 Security Analysis . . . 33
3.1.2 Modified General Profile . . . 37
3.2 Simplified Profile . . . 45
3.2.1 Security Analysis . . . 46

IV RANDOMNESS GENERATION IN KERBEROS . . . 55

4.1 PRG from Iterates . . . 55
4.2 Our PRG Construction . . . 57
4.2.1 The Scheme . . . 58
4.2.2 Security . . . 59
4.3 Proof of Theorem 4.2.2 . . . 59
4.4 Relaxing the Regularity Assumption . . . 66
4.5 Efficiency Improvement . . . 68

V REVOCATION IN NEWER TYPES OF ENCRYPTION . . . 69

5.1 Revocable IBE and its Security . . . 69
5.1.1 Syntax of Revocable IBE . . . 69
5.1.2 Security of Revocable IBE . . . 71
5.2 Our Main Construction . . . 73
5.3 Addressing CCA Security . . . 89
5.4 Revocable ABE and Fuzzy IBE . . . 102

VI CONCLUSIONS . . . 104

REFERENCES . . . 106

VITA . . . 113

LIST OF TABLES

1 Key Update Complexity (# of users n, # of revoked users r) 82

LIST OF FIGURES

1 Birthday Attack (function $f : \{0,1\}^m \rightarrow \{0,1\}^n$, # of trials q) 27

2 Encode-then-Checksum-then-Encrypt Paradigm 32

3 Encode-then-Encrypt&MAC Paradigm 46

4 Pictorial Description of KUNodes Function 76

SUMMARY

Kerberos is a widely-deployed network authentication protocol that is being considered for standardization. Like other standard protocols, Kerberos is no exception to security flaws and weaknesses, as has been demonstrated in several prior works. Provable security guarantees go a long way in restoring users' faith, thus making a protocol an even stronger candidate for standards. In this thesis, our goal was thus to provide provable security support for Kerberos and other practical protocols. Our contributions are three-fold:

1. We first look at the symmetric encryption schemes employed in the current version 5 of Kerberos. Several recent results have analyzed a significant part of Kerberos v.5 using formal-methods-based approaches, which are meaningful only if the underlying encryption schemes satisfy strong cryptographic notions of privacy and authenticity. However, to our knowledge these schemes were never analyzed and proven to satisfy such notions. This thesis aims to bridge this gap. Our provable security analyses confirm that some of the encryption scheme options in Kerberos v.5 already provide privacy and authenticity, and for the remaining we suggest slight modifications for the same.

2. We next turn our attention to the ways in which the keys and other random strings needed in cryptographic schemes employed by practical protocols are generated. Randomness needs to be carefully generated for the provable security guarantees to hold. We propose an efficient pseudorandom generator (PRG) based on hash functions. The security of our PRG relies on exponential collision-resistance and regularity of the underlying hash function. Our PRG can be used to generate various strings, like session keys, sequence numbers, confounders, etc., which are all *suggested* to

be generated randomly in the Kerberos v.5 specification, but no algorithms are mentioned. Each of the above strings are required to satisfy different properties, all of which are trivially satisfied by the pseudorandom strings output by a PRG.

3. Finally, we look at the problem of revocation associated with two relatively new types of encryption schemes: identity-based encryption (IBE) and attribute-based encryption (ABE). While these encryption schemes are relatively less efficient compared to public-key encryption schemes, they have already been used (and are very likely to be used in future, as well) in many practical protocols due to their attractive features. Any setting, public-key, identity-based, or attribute-based, must provide a means to revoke users from the system. However, unlike public-key encryption, there has been little prior work on studying the revocation mechanisms in an IBE or ABE. We propose new primitives and their efficient and provably secure instantiations, focusing on the revocation problem.

We would like to note that even though all the results presented in this thesis are motivated mainly by provable security in practice, only the first bullet above has a direct impact on a practical and widely deployed protocol Kerberos. Our PRG is the most efficient construction among theoretical PRGs, but it may still not be efficient enough to be directly usable in practical protocols. And our results and techniques for revocation in IBE and ABE have found much wider applications in information security, such as mobile social networks, cloud-based secure health records, data outsourcing systems, vehicular ad-hoc networks, etc.

CHAPTER I

INTRODUCTION

Kerberos is a trusted third party network authentication protocol that allows a client to authenticate herself to multiple services, e.g., file servers and printers, with a single login. Kerberos has become widely deployed since its origination as MIT's project Athena in 1988. It has been adopted by many large universities and corporations, is part of all major computing platforms, e.g., Windows (starting from Windows 2000), Linux, UNIX, and Mac OS X, and is a draft standard at IETF [92].

Provable security was introduced in the pioneering work of Goldwasser and Micali [50], and it provides us with a sound, mathematical framework for either designing new schemes/protocols, or analyzing existing ones for security guarantees. In the provable security approach, the very first step is to define a security model for a given security goal, such as achieving privacy via encryption. A security definition specifies precisely what an adversary is allowed to do and what it means to break the scheme. The next step is the design and/or analysis of a scheme. This may include, depending on the scheme, either a proof (by reduction) that the scheme is secure, or an attack showing some insecurity, both following the security model defined. A proof by reduction shows that if an adversary can break the scheme, then one can actually break the security of the building blocks, which are assumed to be secure. The final step is to justify the assmptions, if necessary.

Many popular and widely deployed protocols (or, the cryptographic schemes employed in them) have been analyzed by cryptographers using the provable security approach, for example, Secure Shell (SSH) [15], Wired Equivalent Privacy (WEP) [23], and Internet

Protocol Security (IPsec) [79, 38, 39], to name a few. These analyses normally reveal important security weaknesses and design flaws in the protocols, and are most of the time accompanied with provably secure ways to fix them. However, to the best of our knowledge, no such provable security analysis was performed on Kerberos, which made us wonder if Kerberos was really fit to be a *standard* network authentication protocol. Several attacks [22, 89, 9, 8, 86, 93] on the previous versions of the protocol indicated otherwise.

The main motivation of this thesis was thus to provide provable security support for Kerberos and other practical protocols. Our contributions are three-fold, and we present them in the following three sections.

1.1 Symmetric Encryption in Kerberos

1.1.1 Motivation

Analyzing a complex protocol like Kerberos is not an easy task. The analysis can be greatly simplified by taking a modular approach: first dividing the protocol into several smaller components, then analyzing each of the components in isolation, and finally analyzing how each of these components interact with each other. This is, however, easier said than done. Formal methods (a.k.a., symbolic approach), which treats cryptographic operations as formal expressions, can be very helpful toward this goal. In formal methods, cryptographic primitives are modeled to have idealized security properties (e.g., a ciphertext reveals absolutely no information about either the key, or the message), and then the whole protocol is analyzed via formal expressions to see if it satisfies the required properties.

Several prior works have analyzed Kerberos using the symbolic approach. Butler et al. [35, 36] have analyzed significant portions of the current version of Kerberos and its extensions in the symbolic approach (i.e., Dolev-Yao model [41]), and have formally verified that the design of Kerberos' current version meets the desired goals for the most part. However, a known limitation of such analyses is a high level of abstraction. A significant advance

was made by a recent work by Backes et al. [4], in that it is the only work providing symbolic analysis that also guarantees security in the computational setting, the well-accepted strongest model of security. Their results use the computational soundness model due to Backes et al. [6, 7, 5]. However, for their results to hold, cryptographic primitives used in the protocol need to satisfy strong notions of security (in the computational setting). Namely, as the result of [73, 1] implies, the symmetric encryption schemes utilized by the protocol need to provide privacy against chosen-ciphertext attacks (be IND-CCA secure), as well as authenticity and integrity of ciphertexts (be INT-CTXT secure).

However, it was not known whether *authenticated* encryption schemes[1] in Kerberos are IND-CCA and INT-CTXT secure. Certain known vulnerabilities [93] indicated that encryption schemes in version 4 did not satisfy these notions. While the schemes in the current version, v.5, were designed to resist known attacks, it was not clear whether they *provably* resist all attacks of the class, and if they do, under what assumptions. Our work aims to close this gap.

1.1.2 Related work

Bellare and Namprempre [16] studied various ways to securely compose secure (indistinguishable under chosen-plaintext attacks, or "IND-CPA") encryption and secure (unforgeable against chosen-message attacks, or "UF-CMA") message authentication code (MAC) schemes. They showed that only one out of the three most straightforward composition methods, Encrypt-then-MAC, is secure in general, i.e. always yields an IND-CCA and INT-CTXT encryption scheme. At the same time, certain secure components can yield a scheme, constructed via Encrypt-and-MAC or MAC-then-Encrypt paradigms, that is not IND-CCA or not INT-CTXT. If Kerberos' design had utilized the Encrypt-then-MAC composition method with secure encryption and MAC schemes, we would have nothing to

[1] We will often refer to encryption schemes whose goal is to provide both privacy and authenticity as *authenticated* encryption.

prove here. However, Kerberos uses some variations of Encrypt-and-MAC and MAC-then-Encrypt methods that also employ some encodings on the message, i.e. preprocessing of the message before encryption and MAC are applied.

Bellare et al. [15] analyzed the security of encryption in another widely deployed protocol, Secure Shell (a.k.a. SSH). They suggested several modifications to the SSH encryption to fix certain flaws, and they proved that the resulting scheme is IND-CCA and INT-CTXT secure. They also provided general results about security of stateful encryption schemes composed according to the Encode-then-Encrypt-and-MAC paradigm, assuming certain security properties of the base encoding, encryption, and MAC schemes. The encryption scheme proposed for the revision of Kerberos v.5 (cf. Simplified Profile in [82]) conforms to the Encode-then-Encrypt-and-MAC method. However, the security results from [15] do not directly imply strong security for Simplified Profile in Kerberos. First, the general results from [15] do not guarantee a strong notion of integrity of ciphertexts; they only consider a weaker notion of integrity of plaintexts. Second, the result of [15] requires an IND-CPA secure base encryption scheme, but the base encryption in Kerberos is CBC with fixed IV, which is not IND-CPA secure.

Krawczyk [66] showed that the MAC-then-Encrypt composition method yields a secure authenticated encryption scheme, if the underlying MAC is UF-CMA, and if the encryption scheme uses a PRF blockcipher in CBC with random IV mode. We cannot use this result to prove security of our suggested encryption scheme that we call "Modified General Profile", because the latter uses the CBC mode with *zero* IV, and moreover, it uses a particular encoding scheme so that the confounder (that basically plays the role of random IV for CBC) is also being MACed and encrypted. Proving this scheme requires special care.

Accordingly, we needed to analyze the authenticated encryption schemes in Kerberos from scratch.

1.1.3 Our Contributions

We take a close look at the encryption schemes used in Kerberos v.5 (according to its specifications [81, 82]), in order to prove them secure, in the IND-CCA and INT-CTXT sense, assuming the underlying building blocks (e.g., a blockcipher) are secure. Our results complement the formal methods-based analysis of Kerberos as a key establishment and authentication protocol [35, 36].

General Profile. We first look at the encryption scheme description in the current version, v.5, specification (cf. [82], Section 6). We will refer to it as "General Profile". Fix a blockcipher with input-output length n, and a key for it; a checksum, i.e., a hash function with arbitrary input length, and output length l. A message M is first padded to make the length of the message plus l, a multiple of n. Next, a random n-bit string $conf$ is chosen. Then the checksum is applied to the string $conf||0^l||M$. Let us call the checksum's output σ. Finally, the blockcipher in the CBC mode with fixed initial vector $IV = 0^n$ is applied to the string $conf||\sigma||M$. Decryption is defined accordingly. Security of the scheme depends on how the checksum function is instantiated. The suggested instantiation is a hash function. We observe that General Profile conforms to a general Encode-then-Checksum-then-Encrypt construction, and the latter has a weakness. Namely, we show that even if one assumes the "more secure" component options, e.g., a secure blockcipher in a secure encryption mode and a secure hash function, the Encode-then-Checksum-then-Encrypt construction is not secure *in general*. That is, there exist attacks on the scheme composed of certain secure components, which shows that it does not provide integrity of ciphertexts. We note that these attacks do not apply to General Profile itself, as it uses a particular encryption scheme recommended in [82]. Nevertheless, the attacks show a weakness in the overall design.

Modified General Profile. We propose simple and easy to implement modifications that are sufficient for proving the security of the design of General Profile. Namely, we show

that if the scheme uses a secure blockcipher (a pseudorandom function, or "PRF") in the CBC mode, as specified by General Profile, and if a message authentication code (MAC) that is also a PRF is used as a checksum in place of the hash function, then Modified General Profile yields an encryption scheme that is IND-CCA and INT-CTXT secure. In particular, AES that is assumed to be a PRF and HMAC that is proven to be a PRF assuming the underlying compression hash function is a PRF [11, 10], are good candidates for a blockcipher and MAC, respectively.

SIMPLIFIED PROFILE. Next, we look at the recently proposed revisions to the encryption design in Kerberos, known as Simplified Profile (cf. Section 5 in [82] and [81]). This encryption scheme, for which implementations have not caught up yet, recommends to use AES or Triple-DES as a blockcipher, and HMAC [11] as a MAC, in the following manner. The message is first encoded such that the necessary padding is appended, and a random confounder (the name was suggested in most Kerberos specifications) is prepended. The blockcipher in CBC mode or a variant of CBC mode with ciphertext-stealing[2], both with fixed all-zero-bit IV, and HMAC are applied to the encoded message independently to yield two parts of the resulting ciphertext. Decryption is defined accordingly. We prove that this method yields an encryption scheme that is IND-CCA and INT-CTXT secure, under the assumption that both the blockcipher and the MAC are PRF. This confirms soundness of the design of Simplified Profile. AES is conjectured to be a PRF, Triple DES was shown to be a PRF in the ideal cipher model [21], and as mentioned before, HMAC was proven to be a PRF, assuming the underlying compression hash function is a PRF [10]. Therefore, they are the right choices of instantiations for Simplified Profile. We note that even though Simplified Profile uses the CBC scheme with a fixed IV, this does not compromise the security because pre-pending a random confounder to the message before encrypting makes the scheme equivalent to the CBC with random IV.

[2]Even though we analyze Simplified Profile only with "plain" CBC mode of encryption, we note that our analysis can easily be extended to CBC mode with ciphertext-stealing, and the results remain unaffected.

While our results are not as unexpected or "catchy" as some results discovering a flaw or implementing an attack on a practical protocol, they are far from being less important. Having provable security guarantees is an invaluable benefit for any cryptographic design, especially a widely deployed protocol. Our results together with the formal methods-based results in the symbolic setting constitute strong provable security support for the design of Kerberos.

This is a joint work with Alexandra Boldyreva. An extended abstract version of this work appeared at the 2007 IEEE Symposium on Security and Privacy [27], and a full version appeared in the Journal of IET Information Security [28].

1.2 Randomness Generation in Practical Protocols

1.2.1 Motivation

The Kerberos v.5 specification [92] suggests that various strings, e.g., session keys, sequence numbers, confounders, etc., be generated randomly[3]. While it does not specific any algorithm for generating those strings, it does mention certain requirements, such as (1) it should be impossible to guess the next session key based on the knowledge of past session keys, (2) a sequence number should not collide with other sequence numbers currently in use, and so on.

Rather than trying to satisfy the different specific requirements of any protocol, there is a unified approach which trivially satisfies all such requirements: use a pseudorandom generator (PRG) to generate the desired amount of pseudorandom bits (which can't be distinguished from perfectly random bits by any polynomial-time machine) from a small amount of perfectly random bits, called as the seed of the PRG, and then use the pseudorandom bits as session keys, sequence numbers, or any such strings that are suggested to be generated randomly.

[3]The term "random" is used loosely here, and doesn't necessarily mean perfectly random, where all the outcomes are equally likely.

Protocol analyses done using symbolic approach assume perfect randomness. However, if we only care about real-world (i.e., computationally bounded) adversaries, then *pseudo*randomness is just as good, because as pointed out above, for a polynomial-time machine pseudorandom strings are equivalent to perfectly random strings.

PRG is an important cryptographic primitive that was introduced by Blum and Micali [24], and later formalized into its current form by Yao [91]. PRGs can also be used as building blocks for more complex cryptographic objects like pseudorandom function (PRF) [47], bit commitment [75], etc.

In their seminal work, Håstad et al. [59] building on the previous works [61, 58] show how to construct a PRG, henceforth called HILL-PRG, from any one-way function. While the construction is of great theoretical value, it is extremely (orders of magnitude) inefficient compared to Blum-Micali-Yao (BMY) PRG that builds on a one-way *permutation*. BMY-PRG is the most efficient known construction, whose security relies on a reasonable assumption.

Practical, standardized PRGs [3, 42, 40] can be classified mainly into two categories, depending on the cryptographic primitives they are based on, namely, block-ciphers and hash functions[4]. These PRGs are not our focus for the following reasons:

- The security proofs of block-cipher-based PRGs rely on idealised models of computation, like ideal cipher model.

- The security proofs of hash-function-based PRGs assume that the underlying hash function is a pseudorandom function family (PRF). A PRF is a collection of efficiently-computable functions, such that a function chosen at random from this collection can't be distinguished by any polynomial-time machine from a random oracle (i.e., a function whose outputs are fixed completely at random). We believe that this assumption on a hash function is quite unreasonable, as hash functions not only do not

[4]By a hash function, we mean a function whose range is smaller than the domain, also referred to as a compression function.

have secret keys, but are usually keyless.

We investigate a question of finding an *efficient* hash-function-based PRG, whose security relies on *collision-resistance*, a very well-studied and widely-used property of a hash function. A collision-resistant hash function (CRHF) is of course one-way but certainly not a permutation, as it compresses the input, and hence BMY-PRG is not suitable for our problem.

1.2.2 Related Work

As we will explain soon, the seed length (as a function of the input length m of the underlying function) is an important measure of the efficiency and the security of a PRG. The best known bound for HILL-PRG of $O(m^8)$ was shown by Holenstein [60]. This was later improved (for an alternative construction) to $O(m^7)$ and $O(m^4)$ by Haitner et al. in [55] and [56], respectively. While we certainly don't want longer seeds for obvious reasons, they also have a great impact on the security reduction of the PRG, in that it gives a lower bound on the security, i.e., for the same level of security, HILL-PRG would require a seed that is of size at least an eighth power of the seed size required by BMY-PRG. We present an example to truly appreciate the effect of seed length on the security of a PRG. Say, we have a one-way function that is secure, according to current standards, only for inputs of size at least 128 bits, then Holenstein's proof shows that HILL-PRG is secure only for seeds of size (ignoring constants) at least 2^{56} bits! Several works have tried to bridge this huge gap from BMY-PRG's seed length of $O(m)$, by making stronger assumptions on the underlying function. Following are the two main types of strengthening in the assumption:

- **Regularity.** Goldreich et al. [48] gave a construction of PRG with seed length $O(m^3)$, whose security requires that the underlying function is one-way and *regular*. This was later improved by Haitner et al. [55], where they first present a tighter security proof for a construction similar to that of Goldreich et al., thus improving the seed length to $O(m^2)$ (cf. Section 3.3 in [55]). In the following section of the same work,

Haitner et al. show how the seed length can be further reduced to $O(m \log m)$ by the use of bounded-space generators of Nisan [77] (or, Impagliazzo et al. [62]).

- **Exponential hardness.** Holenstein [60] gave a construction of PRG with seed length $O(m^5)$, whose security relies on the underlying function being an *exponentially hard* one-way function. This was later improved by Haitner et al. to seed length $O(m^2)$ in [54] and [56], where the latter (unlike prior works) doesn't require adaptive calls to the one-way function.

1.2.3 Our Contributions

We construct a new hash-function-based PRG with seed length less than $2m$, i.e., as efficient as BMY-PRG, thus improving the efficiency over all prior works which do not rely on permutations (i.e., function-based PRGs) and have reasonable assumptions. Our scheme is reminiscent of the classical constructions [24, 91] iterating a function on a random seed and extracting Goldreich-Levin hardcore bits [49] at each iteration step. One notable difference from BMY-PRG is that instead of a permutation, we use a hash function. Let h be a hash function mapping strings of size m bits to strings of size n bits, for $m > n$. Assume we have a random seed (x, r), where both x and r are n bits long, and we want to generate l ($> 2n$) pseudorandom bits. The first bit of the output is the inner product of x and r, denoted as $\langle x, r \rangle$. To generate the second bit, compute the first hash-iterate $h^1(x) \leftarrow h(x||0^{m-n})$, and output $\langle h^1(x), r \rangle$. For the third bit, compute the second hash-iterate $h^2(x) \leftarrow h(h^1(x)||0^{m-n})$, and output $\langle h^2(x), r \rangle$. Repeat this process until $(l - n)$ bits are output, and also output r.

The latest PRG of this type that relies on reasonable assumptions (regularity and one-wayness) is due to Haitner et al. [55]. In addition to a regular one-way function, each iteration in their scheme uses a new pairwise-independent function (which is basically the only main difference from our construction), whose descriptions are part of the seed. Our construction presented above does not use pairwise-independent functions and is thus more efficient, requiring less computation and a significantly shorter seed. Our scheme's security

relies on the standard notions of collision-resistance and regularity of the underlying hash function, where the collision-resistance is required to be *exponential* (such a function is also referred in the literature as an "exponentially hard CRHF"). In particular, any polynomial-time adversary should have less than $2^{-n/2}$ probability of finding collisions, where n is the output size of the hash function. This should not be confused with the famous birthday bound, which roughly says that with $2^{n/2}$ number of random trials, one can find collisions (with noticeable probability) in any hash function of output size n. Here, we are talking about the probability of collision and not the number of trials.

To the best of our knowledge, this is the first attempt to combine the above two strengthenings (i.e., regularity and exponential hardness) for improving the efficiency of a function-based PRG. Exponential collision-resistance is not a new assumption, but only requires a more strict upper bound on the probability of finding collisions, and unlike the pseudorandomness of hash functions (which not only do not use secret keys, but are usually keyless), it is still a very well accepted assumption in the community. Also, given the search for a new hash standard SHA-3 by NIST [78], it is plausible that some (if not all) of the candidate submissions to the competition provide exponential collision-resistance. We later show how to relax the regularity assumption by introducing a new notion that we call *worst-case regularity*. The notion of worst-case regularity lower bounds the size of the smallest set of preimages of different elements in the range, while the common regularity assumption requires all such sets to be of equal size. It was shown by Bellare and Kohno [14] that collision-resistance degrades exponentially (in the range of the function) when a function deviates from regularity, so a CRHF must be very "close" to regular, and experiments on practical hashes like SHA-1 support this claim (cf. Section 11 in [14]). So, the worst-case regularity assumption on a practical CRHF seems to be reasonable as well. We note that a notion similar to ours, called "weakly regular" was introduced in [48]. This notion doesn't seem to be useful for our proof, because at a high level it captures the average of the sizes of different preimage sets of a function, whereas what we need is a lower bound on these

sizes.

Levin [67] observed that the BMY-type constructions are secure for functions that are one-way even when applied on their own outputs, a property called *one-way on iterates* (OWI), which one-way permutations trivially satisfy. However, it would be a stretch to assume that practical hashes have this property. We also note that collision-resistance alone may not be sufficient to prove that a function has the OWI property. Consider a CRHF h that acts as a permutation after one application, i.e., for any x in the domain of h, $h(h(x))$ is a permutation on $h(x)$ (some padding can be used to make $h(x)$ of input-size, we omit this padding here for simplicity). For such a CRHF, a security reduction from OWI to collision-resistance is not possible. The reason is that the output of an adversary that can break the OWI security $\left(y \in h^{-1}(h(h(x)))\right)$ cannot be used to find collisions in h, because the set $h^{-1}(h(h(x)))$ has just one element due to h being a permutation after one application. Someone familiar with the proofs of BMY and related PRGs may also be skeptical about the other direction, i.e., proving the security of our scheme assuming only the regularity and collision-resistance of h, without employing the "re-randomizing" pairwise-independent functions. The reason is that the security requires h to remain one-way on every iteration, but while h is believed to be collision-resistant and thus one-way (i.e., it is hard to invert $h(x)$ for a random point x in the domain), it is not necessarily hard to invert $h(h(x))$, because $h(x)$ (for a random x) is not necessarily a random point in the domain. In other words, the sets of points to which h is applied may shrink with each iteration, diminishing the one-wayness property of h, and thus violating the security of the PRG. Somewhat surprisingly, we show that these sets in our construction do not shrink significantly, if it is exponentially hard to find collisions in h. Unlike previous results on the security of PRGs, our theorem provides a concrete security statement, so that it is possible to see exactly how the security of our PRG degrades with the degradation in the collision-resistance of the underlying hash function, and thus allows a more accurate comparison with other schemes.

Our construction is very efficient (though still not comparable to practical, standardized

PRGs [42]) and simple, as at each iteration it uses a hash function and an inner-product computation, both of which are relatively fast. In Section 4.5, we show how using a classical method of [48, 46] the efficiency of our scheme can be further improved by extracting up to a constant fraction of n hardcore bits at each iteration, as the underlying CRHF is assumed to be exponentially hard. We recall that our scheme is similar to the basic construction (that doesn't use bounded-space generators and has a seed length of $O(m^2)$) of [55], but we do not use pairwise-independent functions, which permits significant efficiency improvements, allowing our scheme to have a very short seed. To put the comparison in perspective, the basic scheme of [55] implemented with the compression function of SHA-256 (as the regular one-way function) would require around half a million random bits (as seed) to generate one extra pseudorandom bit, while our construction would just require 512 bits. Our security reduction is very tight, comparable to that of [55], even though the latter does not provide all the details for the concrete security of their PRG. While our construction is mainly of theoretical interest, we believe our approach and treatment has moved theoretically sound PRGs much further towards practical use. The novel worst-case regularity definition may be of independent interest.

This is a joint work with Alexandra Boldyreva. An extended abstract version of this work appeared at the 2012 Cryptographers' Track of the RSA Conference (CT-RSA) [29], and a full version is available from IACR's Cryptology ePrint Archive [30].

1.3 Revocation in Newer Types of Encryption

1.3.1 Motivation

Identity (ID)-based encryption, or IBE for short, and its recent cousin attribute-based encryption (ABE) are exciting alternatives to public-key encryption, which eliminate the need for a Public Key Infrastructure (PKI) that makes publicly available the mapping between identities, public keys, and validity of the latter. The senders using an IBE (or, ABE) do

not need to look up the public keys and the corresponding certificates of the receivers, because the identities, e.g., emails or IP addresses (or, in case of an ABE, a set of attributes) together with common public parameters are sufficient for encryption. The private keys of the users are issued by a trusted third party called the private key generator (PKG). Ideas of identity-based cryptography go back to 1984 and Shamir [84], but the first IBE scheme was constructed by Boneh and Franklin only in 2001 [34], building on the progress in elliptic curves with bilinear pairings. The first ABE scheme under the name "Fuzzy IBE" came soon after in 2005 [83].

Any setting, public-key, identity-based, or attribute-based, must provide a means to revoke users from the system, e.g., if their private keys get compromised. In a PKI setting, a certification authority informs the senders about expired or revoked keys of the users via publicly available digital certificates and certificate revocation lists.

As a solution to this problem for IBE, Boneh and Franklin [34] suggested that users renew their private keys periodically, e.g., every week, and senders use the receivers' identities concatenated with the current time period, e.g., "week 17 of 2012". Notice that since only the PKG's public key and the receiver's identity are needed to encrypt, and there is no way to communicate to the senders that an identity has been revoked, such a mechanism to regularly update users' private keys seems to be the only viable solution to the revocation problem. This means that all users, regardless of whether their keys have been exposed or not, have to regularly get in contact with the PKG, prove their identity and get new private keys. The PKG must be online for all such transactions, and a secure channel must be established between the PKG and each user to transmit the private key. Taking scalability of IBE deployment into account, we observe that for a very large number of users this may become a bottleneck.

We note that alternatively, in order to avoid the need for interaction and a secure channel, the PKG may encrypt the new keys of non-revoked users under their identities and the previous time period, and send the ciphertexts to these users (or, post them online). With

this approach, for every non-revoked user in the system, the PKG is required to perform one key generation and one encryption operation per key update. We note that this solution, just as the original suggestion, requires the PKG to do work linear in the number of users, and does not scale well, as the number of users grows. Our goal was thus to study this problem and find solutions to alleviate it.

While IBEs and ABEs are relatively less efficient compared to public-key encryption schemes, they have already been used (and are very likely to be used in future, as well) in many practical protocols due to their attractive features. Also, the techniques and solutions presented in this thesis have found applications in securing personal health records in cloud computing [68], encryption-based access control in social networks [63], securing content-centric network [94], user revocation scheme in mobile social networks [69], and public key management for vehicular ad hoc networks [85], to name a few.

1.3.2 Related work

Efficient revocation is a well-studied problem in the traditional PKI setting, e.g., [71, 76, 2, 74, 72, 45, 51]. However in the setting of IBE, there has been little work on studying the revocation mechanisms. Hanaoka et al. [57] propose a way for the users to periodically renew their private keys without interacting with the PKG. The PKG publicly posts the key update information, which is much more convenient. However, each user needs to posses a tamper-resistant hardware device. This assumption makes the solution rather cumbersome.

Revocation has been studied in the ID-based setting with mediators [33, 70]. In this setting, there is a special semi-trusted third party called a mediator who holds shares of all users' private keys, and helps users to decrypt each ciphertext. If an identity is revoked, the mediator is instructed to stop helping the user. However, we want to focus on a much more practical, standard IBE setting where users are able to decrypt on their own.

The main goal of a broadcast encryption is to prevent revoked users from accessing secret information being broadcast. The broadcast encryption solutions, however, and in

particular ID-based broadcast encryption ones, do not directly translate into solutions for our problem. In a broadcast encryption, a non-revoked user can help a revoked user gain access to the encrypted message being broadcast (as the message is the same for all parties). On the other hand, in the IBE setting a revoked user, or the adversary holding its private key, should not be able to decrypt encrypted messages even if it colludes with any number of non-revoked users.

Thus, to the best of our knowledge, the solution proposed by Boneh and Franklin in [34] was the most practical user revocation solution in the IBE setting.

1.3.3 Our Contributions

We propose a new way to mitigate the limitation of IBE with regard to revocation, and improve efficiency over the prior solution. We want to remove interaction form the process of key update, as keeping the PKG online can be a bottleneck, especially if the number of users is very large. At the same time, we do not want to employ trusted hardware, and we want to significantly minimize the work done by the PKG and users.

First we define the Revocable IBE primitive and its security model that formalizes the possible threats. The model, of course, takes into account all adversarial capabilities of the standard IBE security notion. I.e., the adversary should be able to learn private keys of users with identities of its choice, and in the case of chosen-ciphertext attack, to also see decryptions of messages (of its choosing) encrypted under the challenge identity. The adversary should not be able to learn any partial information about the messages encrypted for the challenge identity. In addition, we consider the adversary having access to periodic key updates (as we assume this information is public), and being able to revoke users with identities of its choice. The adversary should not be able to learn any partial information about the messages encrypted for any revoked identity, if the message is encrypted after the time of revocation (i.e., for the identity containing the time past the revocation time).

We show that it is possible to reduce the amount of work a PKG has to do for key

updates and the total size of key updates to *logarithmic* in the number of users, while keeping the key update process non-interactive, and encryption and decryption efficient.

Our idea is to build on the Fuzzy IBE construction by Sahai and Waters [83]. The Fuzzy IBE primitive provides some sort of error-tolerance, i.e., identities are viewed as sets of attributes, and a user can decrypt if it possesses keys for enough of (but not necessarily all) attributes a ciphertext is encrypted under. At the same time, colluding users cannot combine their keys to decrypt a ciphertext which none of them were able to decrypt independently.

We propose to combine the Fuzzy IBE construction from [83] with the binary tree data structure, which was previously used to improve the efficiency of revocation mechanisms in the PKI setting [76, 2]. In order to decrypt a ciphertext encrypted under an identity and time period, the user must possess the keys for these two attributes. The PKG publicly posts and regularly updates the keys for the current time attribute. Even though the time attributes are the same for all users, this does not have to compromise security, thanks to the collusion-resistance property of Fuzzy IBE. To reduce the size of key updates from linear to logarithmic in the number of users, the binary tree data structure is used. Here, we employ a trick to modify the Fuzzy IBE scheme in such a way that collusion of some users (corresponding to non-revoked users in our scheme) on some attributes (i.e., time attribute) is possible. We provide more details and present the full construction in Section 5.2.

While our scheme provides major computation and bandwidth efficiency improvements at the stage of key update, it also permits efficient encryption and decryption. We show that our scheme *provably* guarantees security assuming the decisional bilinear Diffie-Hellman problem is hard, which is a quite common assumption nowadays (cf. e.g., [31, 83, 88, 53]).

We also show two ways to address chosen-ciphertext attacks. Our first solution is to modify our scheme by additionally employing a strongly-unforgeable one-time signature scheme in a manner somewhat similar to [37, 53]. We also show that it is possible to employ the Fujisaki-Okamoto (FO) transform [43, 44]. Security of the latter solution relies on the random oracle model [18], but unlike the former solution, it is generic, in that it can

be applied to any Revocable IBE scheme.

Finally, we note that the problem of revocation is equally important for ABE schemes. While the same periodic key update solution due to Boneh and Franklin applies, it similarly limits scalability. We show that it is possible to extend our techniques to provide efficient non-interactive key update to Fuzzy IBE [83] and Key-Policy ABE [53] schemes.

This is a joint work with Alexandra Boldyreva and Vipul Goyal. A preliminary version of this work appeared at the 2008 ACM Conference on Computer and Communications Security [25], and a full version is available from IACR's Cryptology ePrint Archive [26].

1.4 Organization and Conclusion

We first present our notations, and recall some definitions and results from prior works in Chapter 2.

In Chapter 3, we present provable security analyses of the two types of authenticated encryption schemes used in Kerberos version 5, called General Profile and Simplified Profile. Our analyses complement the recent formal-methods-based symbolic analyses, and together these results provide strong security guarantees for Kerberos that we believe will help its standardization.

In Chapter 4, we present our hash-function-based PRG, which is very efficient (though still not comparable to practical, standardized PRGs) and simple, as at each iteration it uses a hash function and an inner-product computation, both of which are relatively fast. While our construction is mainly of theoretical interest, we believe our approach and treatment has moved theoretically sound PRGs much further toward practical use.

In Chapter 5, we present our last result solving the problem of efficient revocation in IBEs and ABEs. While these encryption schemes are relatively less efficient compared to public-key encryption schemes, they have already been used (and are very likely to be used in future, as well) in many practical protocols due to their attractive features. Our techniques and solutions have found applications in several varied areas of information

security.

Finally, we conclude in Chapter 6.

CHAPTER II

PRELIMINARIES

2.1 Notation

Let $\mathbb{N}$ denote the set of natural numbers, and $\mathbb{R}$ denote the set of real numbers. We denote by $\kappa \in \mathbb{N}$ our security parameter, and by $n(\cdot)$ a polynomial in κ. We say a function is negligible, if it decreases faster than any polynomial. For any $n \in \mathbb{N}$, let $\mathbb{Z}_n$ denote the set of integers modulo n, and $\mathbb{Z}_n^*$ denote the set $\mathbb{Z}_n \setminus 0$. We denote by $\{0,1\}^*$ the set of all binary strings of finite length. If x is a string, then $|x|$ denotes its length in bits, and if $x \in \mathbb{R}$, then $|x|$ denotes its absolute value. If x and y are strings, then $\langle x, y \rangle$ denotes their inner product modulo 2, i.e., $\sum_{i=1}^{m} x_i \cdot r_i \pmod 2$; and $x \,\|\, y$ denotes their concatenation, and we assume that x and y can be efficiently and unambiguously recovered from $x \,\|\, y$. For a string x, whose length is a multiple of $n \in \mathbb{N}$ bits, $x[i]$ denotes the i^{th} block, meaning $x = x[1]\| \ldots \|x[l]$, where $l = |x|/n$, and for all $i = 1$ to l, $|x[i]| = n$. For a string x, and any integers $0 \leq i < j \leq |x|$, $x_{[i]}$ denotes the i^{th} bit of x, and $x_{[i \ldots j]}$ denotes $x_{[i]} \,\|\, \ldots \,\|\, x_{[j]}$. For an integer k and a bit b, b^k denotes the string consisting of k consecutive "b" bits.

We denote by ϕ an empty set. If S is a finite set, then $s \xleftarrow{\$} \mathsf{S}$ denotes that s is selected uniformly at random from S. We will often write $s_1, s_2, \ldots, s_n \xleftarrow{\$} \mathsf{S}$ as a shorthand for $s_1 \xleftarrow{\$} \mathsf{S}; s_2 \xleftarrow{\$} \mathsf{S}; \ldots; s_n \xleftarrow{\$} \mathsf{S}$. If $\mathsf{S} = \{s_1, s_2, \ldots, s_n\}$, then $\{x_s\}_{s \in \mathsf{S}}$ denotes the set $\{x_{s_1}, x_{s_2}, \ldots, x_{s_n}\}$.

If f is a function, then $\mathsf{Im}(f)$ denotes the image set of f, and for any $y \in \mathsf{Im}(f)$, $\mathsf{Preim}(f, y)$ denotes the set of preimages of y under f. For any $a, b \in \mathbb{N}$, if $a < b$, then for simplicity and correctness we define $\binom{a}{b}$ to be 1.

When describing algorithms, $a \leftarrow b$ denotes that a is assigned the value b. If A is a randomized algorithm and $n \in \mathbb{N}$, then $a \xleftarrow{\$} A(i_1, i_2, \ldots, i_n)$ denotes that a is assigned the outcome of running A on input $(i_1, i_2, \ldots, i_n)$; the dollar sign above the arrow is dropped

if A is deterministic. An adversary is an algorithm. By convention, the running-time of an adversary includes that of its overlying experiment. All algorithms are assumed to be randomized and efficient (i.e., polynomial in the input size), and all functions are assumed to be efficiently computable, unless noted otherwise.

2.2 Symmetric Encryption Schemes and their Security

Definition 2.2.1. [Symmetric Encryption Scheme] A *symmetric encryption scheme* $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$, with associated message space MsgSp, is defined by three algorithms:

- The randomized *key generation* algorithm $\mathcal{K}$ returns a secret key K.
- The (possibly) randomized or stateful *encryption* algorithm $\mathcal{E}$ takes as input the secret key K and a plaintext $M \in \mathsf{MsgSp}$, and returns a ciphertext.
- The deterministic *decryption* algorithm $\mathcal{D}$ takes the secret key K and a ciphertext C to return the corresponding plaintext, or a special symbol $\perp$ indicating that the ciphertext was invalid.

The consistency condition requires that $\mathcal{D}_K(\mathcal{E}_K(M)) = M$, for all K that can be output by $\mathcal{K}$, and all $M \in \mathsf{MsgSp}$.

We now recall cryptographic security notions for encryption. The following definition [12] is for data privacy (confidentiality). It formalizes the requirement that even though an adversary may know some partial information about the data, no additional information is leaked.

Definition 2.2.2. [IND-CPA, IND-CCA] Let $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme. For attack type atk $\in$ {cpa, cca}, adversary A, and a bit b, define the experiments $\mathbf{Exp}^{\text{ind-atk-b}}_{\mathcal{SE}}(A)$ as follows. In all the experiments, first a key K is generated by $\mathcal{K}$. Let LR (left-or-right) be a "selector" that on input M_0, M_1, b returns M_b. The adversary A is given access to a *left-right encryption oracle* $\mathcal{E}_K(\mathrm{LR}(\cdot, \cdot, b))$ that it can query on any pair of messages of

equal length in MsgSp. For atk = cca, i.e., in $\mathbf{Exp}^{\text{ind-cca-b}}_{\mathcal{SE}}(A)$, the adversary is also given a decryption oracle $\mathcal{D}_K(\cdot)$ that it can query on any ciphertext that was not returned by the other oracle. The adversary's goal is to output a bit d as its guess of the challenge bit b; the experiment returns d as well. The *ind-atk-advantage* of an adversary A is defined as:

$$\mathbf{Adv}^{\text{ind-atk}}_{\mathcal{SE}}(A) \;=\; \Pr\left[\,\mathbf{Exp}^{\text{ind-atk-1}}_{\mathcal{SE}}(A) = 1\,\right] - \Pr\left[\,\mathbf{Exp}^{\text{ind-atk-0}}_{\mathcal{SE}}(A) = 1\,\right].$$

The scheme $\mathcal{SE}$ is said to be *indistinguishable against chosen-plaintext attacks* or *IND-CPA* (resp., *chosen-ciphertext attacks* or *IND-CCA*), if for every polynomial-time (in the security parameter κ) adversary A, its ind-cpa (resp., ind-cca) advantage is negligible (in κ).

(We use the standard convention that the running time of an adversary is measured with respect to the entire experiment in which it runs. In computing the total length of the queries made to the LR encryption oracle, we only count the length of one of the messages from the pair.)

It is easy to see that IND-CCA security is a stronger notion that implies IND-CPA security.

The following definition [17, 19] is for authenticity and integrity of encryption. It formalizes the requirement that no adversary should be able to compute a new ciphertext which the receiver will deem valid.

Definition 2.2.3. [INT-CTXT] Let $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme. The encryption scheme is said to provide *authenticity, or ciphertext integrity (be INT-CTXT secure)*, if any polynomial-time (in κ) adversary A can be successful in the following experiment only with negligible (in κ) probability, called the *int-ctxt-advantage* of A, $\mathbf{Adv}^{\text{int-ctxt}}_{\mathcal{SE}}(A)$. In the experiment, first a key K is generated by $\mathcal{K}$. The adversary has access to two oracles: encryption oracle $\mathcal{E}_K(\cdot)$ and verification oracle $\mathcal{V}_K(\cdot)$. On input a ciphertext C, $\mathcal{V}_K(\cdot)$ returns 1, if C was not returned by $\mathcal{E}_K(\cdot)$ and $\mathcal{D}_K(C) \neq \perp$. The adversary is successful in the experiment if $\mathcal{V}_K(\cdot)$ ever returns 1.

It has been shown [17] that if an encryption scheme is IND-CPA and INT-CTXT, then it is also IND-CCA.

Theorem 2.2.4. **[[17], Theorem 3.2]** Let $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme. If it is IND-CPA and INT-CTXT secure, then it is also IND-CCA secure. Concretely, for any adversary A attacking the IND-CCA security of $\mathcal{SE}$, that runs in time t, and makes q_e queries to the left-right encryption oracle, and q_d queries to the decryption oracle, totaling μ_e and μ_d bits, respectively, there exist adversaries B and C attacking the scheme's IND-CPA and INT-CTXT security, respectively, such that

$$\mathbf{Adv}^{\text{ind-cca}}_{\mathcal{SE}}(A) \;\leq\; \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}}(B) + 2 \cdot \mathbf{Adv}^{\text{int-ctxt}}_{\mathcal{SE}}(C) \;.$$

Furthermore, B runs in time[1] t, and makes q_e queries to the left-right encryption oracle, totaling μ_e bits, while C runs in time t, and makes q_e queries to the encryption oracle, and q_d queries to the verification oracle, totaling μ_e and μ_d bits, respectively.

2.3 PRFs, MACs, and their Security

A family of functions is a map $E\colon \mathsf{Keys} \times \mathsf{Dom} \to \mathsf{Ran}$, where we regard Keys as the *keyspace* for the function family in that a *key* $K \in \mathsf{Keys}$ induces a particular function from this family, which we denote by $E_K(\cdot)$.

Definition 2.3.1. **[PRF]** Let $E\colon \mathsf{Keys} \times \mathsf{Dom} \to \mathsf{Ran}$ be a function family. Let R be the set of all functions from Dom to Ran. E is called *pseudorandom, or PRF secure*, if any polynomial-time (in κ) adversary A, with access to an oracle that it can query on messages in MsgSp, has negligible (in κ) *prf-advantage* defined as

$$\mathbf{Adv}^{\text{prf}}_{E}(A) \;=\; \Pr\left[\, K \xleftarrow{\$} \mathsf{Keys} \;:\; A^{E_K(\cdot)} = 1 \,\right] - \Pr\left[\, g \xleftarrow{\$} R \;:\; A^{g(\cdot)} = 1 \,\right] .$$

Definition 2.3.2. **[MAC]** A *message authentication code* $\mathcal{MAC} = (\mathcal{K}, \mathcal{T})$ with associated *message space* MsgSp is defined by two algorithms:

[1] The time complexity given in [17] is slightly different due to the difference in convention.

- The randomized *key generation* algorithm $\mathcal{K}$ returns a secret key K.
- The deterministic[2] *tagging* algorithm $\mathcal{T}$ takes as input the secret key K, and a plaintext $M \in \mathsf{MsgSp}$ to return a tag for M.

For a message-tag pair (M, σ), we say σ is a valid tag for M, if $\sigma = \sigma'$, where $\sigma' \leftarrow \mathcal{T}_{\mathsf{K}}(M)$.

The following security definition [13] requires that any polynomial-time (in κ) adversary can forge a valid tag for a new message only with a negligible probability.

Definition 2.3.3. [UF-CMA] Let $\mathcal{MAC} = (\mathcal{K}, \mathcal{T})$ be a MAC scheme. It is called *unforgeable against chosen-message attacks, or UF-CMA secure*, if any polynomial-time (in κ) adversary A can be successful in the following experiment with only a negligible probability, called the *uf-cma-advantage* of A, $\mathbf{Adv}^{\text{uf-cma}}_{\mathcal{MAC}}(A)$. In the experiment, first a random key K is generated by $\mathcal{K}$. The adversary has access to two oracles: tagging oracle $\mathcal{T}_K(\cdot)$ and verification oracle[3] $\mathcal{V}_K(\cdot, \cdot)$. On input a message-tag pair (m, σ), $\mathcal{V}_K(\cdot, \cdot)$ returns 1, if m was not queried to $\mathcal{T}_K(\cdot)$, and $\mathcal{T}_K(m) = \sigma$, otherwise it returns 0. The adversary is successful in the experiment if $\mathcal{V}_K(\cdot, \cdot)$ ever returns 1.

Another (stronger) security definition requires that the output of the MAC is indistinguishable from a random string. The definition below is very similar to the PRF definition for the function family. The only difference is that now a key generation algorithm is used to generate the key.

Definition 2.3.4. [PRF Security for MACs] Let $\mathcal{MAC} = (\mathcal{K}, \mathcal{T})$ be a MAC scheme. Let R be the set of all functions with the same domain and range as $\mathcal{T}$. $\mathcal{MAC}$ is called *pseudorandom, or PRF secure*, if any polynomial-time (in κ) adversary A, with access to

[2] A MAC does not have to be deterministic, but most practical schemes are, so we consider only deterministic MACs.

[3] Since we only consider deterministic MACs, the verification oracle is not necessary. However, we keep it for generality. In computing the total length of the queries made to the verification oracle, we only count the length of message m, and not the message-tag pair (m, σ).

an oracle that it can query on messages in MsgSp, has a negligible (in κ) *prf-advantage* defined as

$$\mathbf{Adv}^{\mathrm{prf}}_{\mathcal{MAC}}(A) \;=\; \Pr\left[\, K \xleftarrow{\$} \mathcal{K} \;:\; A^{\mathcal{T}_K(\cdot)} = 1 \,\right] - \Pr\left[\, g \xleftarrow{\$} R \;:\; A^{g(\cdot)} = 1 \,\right] .$$

We recall the fact that any MAC that is PRF is also UF-CMA.

Theorem 2.3.5. **[[20], Proposition 6.3]** Let $\mathcal{MAC} = (\mathcal{K}, \mathcal{T})$ be a MAC scheme. Then, for any adversary F attacking UF-CMA security of $\mathcal{MAC}$, that runs in time t, and makes q_t queries to the tagging oracle, and q_v queries to the verification oracle, totaling μ_t and μ_v bits, respectively, there exists an adversary G attacking the PRF security of $\mathcal{MAC}$, such that

$$\mathbf{Adv}^{\text{uf-cma}}_{\mathcal{MAC}}(F) \;\leq\; \mathbf{Adv}^{\mathrm{prf}}_{\mathcal{MAC}}(G) + \frac{q_v}{|\mathsf{Ran}_{\mathcal{T}}|} ,$$

where $\mathsf{Ran}_{\mathcal{T}}$ denotes the range of $\mathcal{T}$. Furthermore, G runs in time t, and makes $(q_t + q_v)$ oracle queries, totaling $(\mu_t + \mu_v)$ bits.

2.4 Encoding Schemes and their Security

An encoding scheme is an unkeyed invertible transformation that is used to extend the message with some associated data, such as padding, a counter, random nonce, etc.

Definition 2.4.1. [Encoding Scheme] An *encoding scheme* $\mathcal{EC} = (\mathit{Encode}, \mathit{Decode})$ with associated *message space* MsgSp is defined by two algorithms. The (possibly) randomized or stateful *encoding* algorithm *Encode* takes a message $M \in \mathsf{MsgSp}$, and outputs a pair of messages (M_e, M_t). The deterministic *decoding* algorithm takes M_e and returns a pair (M, M_t), or $(\perp, \perp)$ on error.

For any message $M \in \mathsf{MsgSp}$, let $(M_e, M_t) \xleftarrow{\$} \mathit{Encode}(M)$, and $(M', M'_t) \leftarrow \mathit{Decode}(M_e)$. Then the consistency condition requires that $M = M'$ and $M_t = M'_t$. We note that in the constructions we will consider, M_e is going to be used as an input to the encryption algorithm, and M_t is going to be used as an input to the MAC algorithm.

The following is from [19, 15].

Definition 2.4.2. [Coll-CPA] Let $\mathcal{EC} = (Encode, Decode)$ be an encoding scheme. It is called *collision-resistant against chosen-plaintext attacks, or Coll-CPA*, if any polynomial-time (in κ) adversary A has only a negligible (in κ) success probability, called the *coll-cpa-advantage* of A, or $\mathbf{Adv}^{\text{coll-cpa}}_{\mathcal{EC}}(A)$, in the following experiment. The adversary has access to the encoding oracle $Encode(\cdot)$, and it is considered successful if it ever gets two replies (M_e, M_t) and (M'_e, M'_t), such that $M_t = M'_t$.

2.5 Hash Function Families and their Security

Because of the known difficulties of defining collision-resistance (cf. Section 6.1 in [20]), we follow the standard approach and define hash function families.

Definition 2.5.1. [Hash Function Family] A *hash function family* $\mathcal{H}$ is a collection of functions, where each $h \in \mathcal{H}$ is a mapping from $\{0,1\}^m$ to $\{0,1\}^n$, such that $m > n$. An instance $h \in \mathcal{H}$ may be described by a key which is publicly known.

Definition 2.5.2. [Collision-Resistance and Target Collision-Resistance] Let $\mathcal{H}$ be a hash function family, where each $h \in \mathcal{H}$ is a mapping from $\{0,1\}^m$ to $\{0,1\}^n$. The *collision-resistance* advantage of an adversary C attacking $\mathcal{H}$, $\mathbf{Adv}^{\text{cr}}_{\mathcal{H}}(C)$ is defined as

$$\Pr\left[\, h \xleftarrow{\$} \mathcal{H},\ x, x' \xleftarrow{\$} C(h) :\ x \neq x' \in \{0,1\}^m \bigwedge h(x) = h(x') \,\right].$$

Also, the *target* collision-resistance advantage of an adversary C attacking $\mathcal{H}$, $\mathbf{Adv}^{\text{tcr}}_{\mathcal{H}}(C)$ is defined as

$$\Pr\left[\, h \xleftarrow{\$} \mathcal{H}, x \xleftarrow{\$} \{0,1\}^m, x' \xleftarrow{\$} C(h,x) :\ x' \in \{0,1\}^m \bigwedge x \neq x' \bigwedge h(x) = h(x') \,\right].$$

Definition 2.5.3. [Birthday Attack] The birthday attack on a function $f : \{0,1\}^m \to \{0,1\}^n$ is defined in Figure 1. In this attack, $q \in \mathbb{N}$ points $x_1, ..., x_q$ are picked independently at random from the domain. If any two of these points form a collision for f, then the attack is successful and those two points are returned. We denote the probability of success of the

For $i = 1, ..., q$
$x_i \xleftarrow{\$} \{0,1\}^m$
$y_i \leftarrow f(x_i)$
If $(\exists j : \ j < i \ \bigwedge \ y_i = y_j \ \bigwedge \ x_i \neq x_j)$, return (x_i, x_j).

Figure 1: Birthday Attack (function $f : \{0,1\}^m \rightarrow \{0,1\}^n$, # of trials q)

birthday attack on f by *collision probability*, $\mathbf{CP}(f, q)$. We will slightly abuse the notation sometimes, and use it for function families, where in $\mathbf{CP}(\mathcal{F}, q)$ for a function family $\mathcal{F}$, would mean the collision probability of a function picked at random from $\mathcal{F}$.

Definition 2.5.4. [Regularity] A function $f : \{0,1\}^m \rightarrow \{0,1\}^n$ is said to be *regular*, if every point in the image set of f have equal number of preimages. Bellare and Kohno introduced the notion of a balance measure, denoted $\mu(f)$ (cf. Section 1 in [14]) to measure the regularity of a function: $\mu(f) = 1$ indicates that the function is fully regular, and $\mu(f) = 0$ means fully irregular (an image point has the maximum number of preimages). The collision probability in the birthday attack for q trials, $\mathbf{CP}(f, q) = \binom{q}{2} \cdot 2^{-n\mu(f)}$ (up to constant factors), so the collision-resistance of any function degrades exponentially (in the range of the function) with the decline in its balance. A collision-resistant hash function (CRHF) must therefore have a balance close to 1, and experiments on practical hashes like SHA-1 support this claim (cf. Equation 2, Section 11 in [14]). So, SHA-1 and other hash functions (SHA-256, SHA-512, etc.) can be assumed to be *close* to regular. We introduce a notion that we call *worst-case regularity* in Section 4.4 that also captures this closeness.

Definition 2.5.5. [One-Wayness] Let $\mathcal{F}$ be a family of functions, where each $f \in \mathcal{F}$ is a mapping from $\{0,1\}^m$ to $\{0,1\}^n$. The *one-way* advantage of an adversary I attacking $\mathcal{F}$, $\mathbf{Adv}^{\text{ow}}_{\mathcal{F}}(I)$ is defined as

$$\Pr\left[f \xleftarrow{\$} \mathcal{F}, x \xleftarrow{\$} \{0,1\}^m, x' \xleftarrow{\$} I(f, f(x)) : \ x' \in \{0,1\}^m \bigwedge f(x') = f(x) \right].$$

The one-way advantage of a function f (instead of a function family) can be defined similarly: the adversary is given $f(x)$ for a random x, and it has to return an element $x' \in \{0,1\}^m$ such that $f(x') = f(x)$.

The following relation between target collision-resistance and one-wayness of a function is well-known.

Theorem 2.5.6. [[20], Corollary 5.5] Let $\mathcal{H}$ be a hash function family, where each $h \in \mathcal{H}$ is a mapping from $\{0,1\}^m$ to $\{0,1\}^n$. Then for an adversary I with running time t_I, there exists an adversary C with running time t_C, so that

$$\mathbf{Adv}^{\mathrm{ow}}_{\mathcal{H}}(I) \leq 2 \cdot \mathbf{Adv}^{\mathrm{tcr}}_{\mathcal{H}}(C) + 2^{n-m}, \text{ and } t_C \approx t_I.$$

We now present a more general definition that also captures the one-wayness.

Definition 2.5.7. [Hard to Compute] Let f and g be functions with the same domain $S_m \subseteq \{0,1\}^m$. The *hard-to-compute* advantage of an adversary I attacking (f,g), $\mathbf{Adv}^{\mathrm{htc}}_{f,g}(I)$ is defined as

$$\Pr\left[x \xleftarrow{\$} S_m : \ I(f(x)) \in \mathsf{Preim}(g, f(x)) \right].$$

Note that for any adversary I and any function f, $\mathbf{Adv}^{\mathrm{ow}}_{f}(I) = \mathbf{Adv}^{\mathrm{htc}}_{f,f}(I)$.

Definition 2.5.8. [Hardcore Predicate] Informally, a hardcore predicate is a bit of the input that is hard to predict significantly better than a random guess. Formally, let $g : \{0,1\}^m \to \{0,1\}^n$, $b : \{0,1\}^m \to \{0,1\}$ be two functions, and $a \xleftarrow{\$} \{0,1\}$ be a random bit. The *hardcore predicate* advantage of adversary A, $\mathbf{Adv}^{\mathrm{hcp}}_{g,b}(A)$ is defined as

$$\Pr\left[x \xleftarrow{\$} \{0,1\}^m : A(g(x), b(x)) = 1 \right] - \Pr\left[x \xleftarrow{\$} \{0,1\}^m : A(g(x), a) = 1 \right].$$

Here $b(x)$ is called the *hardcore predicate (or bit)* of $g(x)$. In this paper, we use the general hardcore predicate construction of Goldreich and Levin [49], called the "GL-hardcore bit". For two bitstrings x $(= x_1 \| \ldots \| x_m)$ and r $(= r_1 \| \ldots \| r_m)$, define $b(x,r) = \langle x, r \rangle$, the inner product of x and r modulo 2. The following theorem is from [55], and states (using our notation) the security of the GL-hardcore bit.

Theorem 2.5.9. [Theorem 2.7, [55]] Let f and g be functions with the same domain $S_m \subseteq \{0,1\}^m$. For a random $x \in S_m$ and a random $r \in \{0,1\}^m$, define $\widehat{f}$ as $\widehat{f}(x,r) = (f(x), r)$, and

its GL-hardcore bit b as $\langle z, r\rangle$, where $z \in \mathsf{Preim}(g, f(x))$ is one of the preimages of $f(x)$ under g. Then for an adversary A with running time t_A, there exists an adversary I with running time t_I, so that

$$\mathbf{Adv}^{\mathrm{hcp}}_{\widehat{f},b}(A) \leq 4 \cdot \mathbf{Adv}^{\mathrm{htc}}_{f,g}(I), \quad \text{and } t_I = O\left(m^3 \cdot t_A \cdot \left(\mathbf{Adv}^{\mathrm{hcp}}_{\widehat{f},b}(A)\right)^{-4}\right).$$

2.6 Pseudorandom Generator

Informally, a pseudorandom generator (PRG) is a function that expands a random seed into a longer pseudorandom bit sequence. PRGs were first proposed and constructed by Blum and Micali [24], and Yao [91]. Let $\mathcal{G} : \{0,1\}^m \to \{0,1\}^l$ be a function, so that $l > m$. We say that $\mathcal{G}$ is a secure PRG, if for any polynomial-time (in the security parameter κ) adversary P attacking $\mathcal{G}$, its prg advantage $\mathbf{Adv}^{\mathrm{prg}}_{\mathcal{G}}(P)$ defined as

$$\Pr\left[\, s \xleftarrow{\$} \{0,1\}^m : \; P(\mathcal{G}(s)) = 1 \,\right] - \Pr\left[\, y \xleftarrow{\$} \{0,1\}^l : \; P(y) = 1 \,\right],$$

is negligible (in κ). Here m is the seed length, and l is the number of pseudorandom bits generated.

2.7 Bilinear Maps

Let $\mathbb{G}, \mathbb{G}_T$ be groups of prime order p (so they are cyclic). A *pairing* is an efficiently computable map $\mathbf{e}\colon \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ such that the following two conditions hold:

- Bilinearity: For all $g_1, g_2 \in \mathbb{G}$ and $x, y \in \mathbb{Z}$, we have $\mathbf{e}(g_1^x, g_2^y) = \mathbf{e}(g_1, g_2)^{xy}$.
- Non-degeneracy: For any generator g of $\mathbb{G}$, $\mathbf{e}(g, g)$ is a generator of $\mathbb{G}_T$.

Note that $\mathbf{e}(\cdot, \cdot)$ is symmetric, since $\mathbf{e}(g^x, g^y) = \mathbf{e}(g, g)^{xy} = \mathbf{e}(g^y, g^x)$.

A *bilinear group generator* $\mathcal{G}$ is an algorithm that on input 1^κ returns $\tilde{\mathbb{G}}$ (which is a description of groups $\mathbb{G}$ and $\mathbb{G}_T$, both of order p), the bilinear map $\mathbf{e}$ as defined above, and also p and a generator g of $\mathbb{G}$. There can be numerous such prime order bilinear group generators. We will not specify a particular one but will use it as a parameter to the hardness

assumption that we use for our security proof. The description of a group should specify the algorithms for group operations (multiplication, inverse and pairing), the algorithm for testing group membership, and also the random group element sampling algorithm. We assume that the group elements are uniquely encoded as strings.

Below we recall a hardness assumption related to bilinear maps, called the *decisional bilinear Diffie-Hellman* (DBDH) problem.

Definition 2.7.1. [Decisional Bilinear Diffie-Hellman] Let $\mathcal{G}$ be a prime order bilinear group generator. The *decisional bilinear Diffie-Hellman (DBDH)* problem is said to be hard for $\mathcal{G}$ if for every efficient adversary A its advantage $\mathbf{Adv}^{\text{dbdh}}_{\mathcal{G}}(A)$ defined as

$$\Pr\left[\, \mathbf{Exp}^{\text{dbdh-real}}_{\mathcal{G}}(A) = 1 \,\right] - \Pr\left[\, \mathbf{Exp}^{\text{dbdh-rand}}_{\mathcal{G}}(A) = 1 \,\right]$$

is a negligible function in κ, and where the experiments are as follows:

Experiment $\mathbf{Exp}^{\text{dbdh-real}}_{\mathcal{G}}(A)$

$(\tilde{\mathbb{G}}, p, g) \xleftarrow{\$} \mathcal{G}(1^\kappa)$; $x, y, z \xleftarrow{\$} \mathbb{Z}_p$

$X \leftarrow g^x$; $Y \leftarrow g^y$; $Z \leftarrow g^z$; $W \leftarrow \mathbf{e}(g, g)^{xyz}$

$d \xleftarrow{\$} A(1^\kappa, \tilde{\mathbb{G}}, p, g, X, Y, Z, W)$

Return d

Experiment $\mathbf{Exp}^{\text{dbdh-rand}}_{\mathcal{G}}(A)$

$(\tilde{\mathbb{G}}, p, g) \xleftarrow{\$} \mathcal{G}(1^\kappa)$; $x, y, z, w \xleftarrow{\$} \mathbb{Z}_p$

$X \leftarrow g^x$; $Y \leftarrow g^y$; $Z \leftarrow g^z$; $W \leftarrow \mathbf{e}(g, g)^w$

$d \xleftarrow{\$} A(1^\kappa, \tilde{\mathbb{G}}, p, g, X, Y, Z, W)$

Return d

CHAPTER III

SYMMETRIC ENCRYPTION IN KERBEROS

In this chapter, we present our first set of results: provable security analyses of encryption schemes used in Kerberos v.5. This chapter is organized in two sections. In Section 3.1, we present our analyses of the most widely used encryption in Kerberos, that we call General Profile, and in Section 3.2, we analyze a recently proposed encryption, called Simplified Profile, that implementations are still catching up with.

3.1 General Profile

We first look at the encryption scheme specified in [82]. This document describes several options, but we note that all the choices conform to a general composition method that we outline below (the design is further generalized in [65]).

Construction 3.1.1. [Encode-then-Checksum-then-Encrypt] Let $\mathcal{SE} = (\mathcal{K}_e, \mathcal{E}, \mathcal{D})$, $\mathcal{EC} = (\mathit{Encode}, \mathit{Decode})$, and $\mathcal{CS} = (\mathcal{K}_t, \mathcal{T})$ be an encryption scheme, an encoding scheme, and a checksum (i.e., a hash, or a MAC). Then the *Encode-then-Checksum-then-Encrypt* scheme $\mathcal{SE}' = (\mathcal{K}', \mathcal{E}', \mathcal{D}')$, with the same message space as that of $\mathcal{EC}$, is defined by a tuple of three algorithms as follows:

- $\mathcal{K}'$ runs $\mathcal{K}_e, \mathcal{K}_t$, and returns their outputs $K_e \parallel K_t$.
- $\mathcal{E}'$ on input $K_e \parallel K_t$ and M, first gets the encodings via $(M_e, M_t) \xleftarrow{\$} \mathit{Encode}(M)$. It then computes $\sigma \leftarrow \mathcal{T}_{K_t}(M_t)$, parses M_e as $M_{el} \| M_{er}$ (M_{el} and M_{er} are the left and right parts of M_e, respectively), and returns $C \xleftarrow{\$} \mathcal{E}_{K_e}(M_{el} \parallel \sigma \parallel M_{er})$.
- $\mathcal{D}'$ on input $K_e \| K_t$ and C, computes $M_e \leftarrow M_{el} \| M_{er}$, σ from $(M_{el} \| \sigma \| M_{er}) \leftarrow \mathcal{D}_{K_e}(C)$, decodes $(M, M_t) \leftarrow \mathit{Decode}(M_e)$, computes $\sigma' \leftarrow \mathcal{T}_{K_t}(M_t)$, and returns M, if $\sigma = \sigma'$, and $\perp$ otherwise.

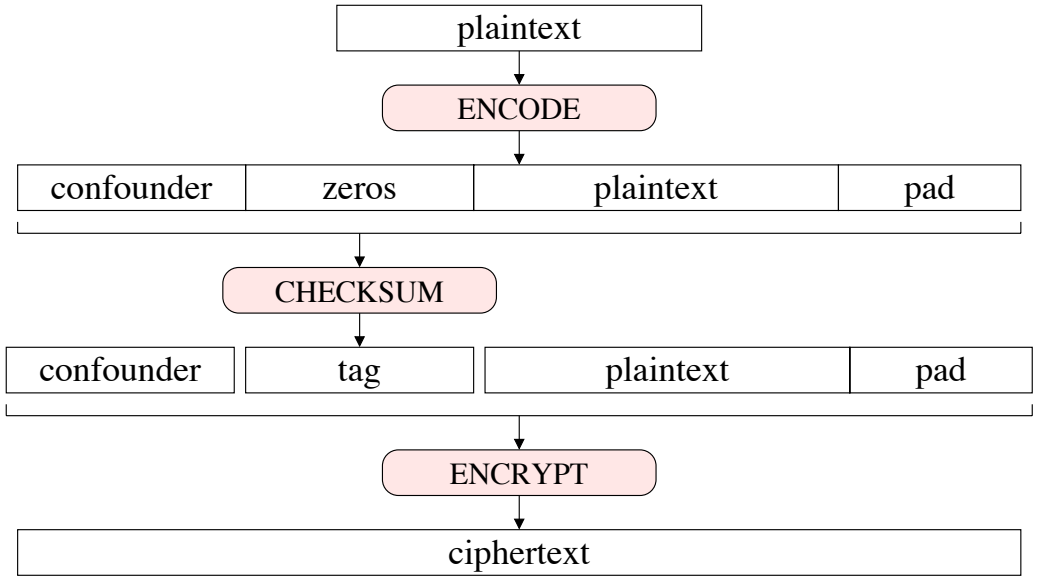

Figure 2: Encode-then-Checksum-then-Encrypt Paradigm

Above, we assume that the outputs of the encoding scheme are compatible with the inputs to $\mathcal{E}$ and $\mathcal{T}$. Figure 2 illustrates the design.

The next construction specifies in more detail how Kerberos' encryption operates, i.e., what specific algorithms instantiate the generic composition method of Construction 3.1.1.

Construction 3.1.2. [Authenticated Encryption in Kerberos: General Profile] Let E: $\{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n$ be a family of permutations, to be referred as a blockcipher. Let $\mathcal{SE} = (\mathcal{K}_e, \mathcal{E}, \mathcal{D})$ be the associated CBC encryption mode (cf. [12] for the formal description), with $IV = 0^n$ [1]. Let $\mathcal{H}$ be a hash function family[2] (to be used as checksum $\mathcal{CS}$), where each $h \in \mathcal{H}$ is a mapping from $\{0,1\}^*$ to $\{0,1\}^l$. Let $\mathcal{EC} = (Encode, Decode)$ be an encoding scheme, such that $Encode$ with $\mathsf{MsgSp} = \{0,1\}^*$, on input M, pads it to make the length of $(l+|M|)$ a multiple of n bits (so that decoding is unambiguous), picks a random

[1]The Kerberos specification also allows stateful update of the IV. In particular, the IV is assigned to be the last block of the previous ciphertext. Our analyses apply to this case as well. But, since this option is not commonly used, we do not consider it in detail. We note, however, that [82] does not specify how the state and IV are updated when the receiver gets an invalid ciphertext. The only reasonable resolution preventing malicious attacks disrupting the future communication may be to issue an error message and reset the IV to 0^n.

[2]This can be keyless, which means that the family contains just one function.

confounder of n bits $conf \xleftarrow{\$} \{0,1\}^n$, computes $M_e \leftarrow conf \,||\, M$, and $M_t \leftarrow conf \,||\, 0^l \,||\, M$, and returns (M_e, M_t). M_{el} is defined to be $conf$, and M_{er} is the rest of M_e. *Decode* on input M_e parses it as $conf||M$, computes $M_t \leftarrow conf||0^l||M$, and returns (M, M_t). Then Construction 3.1.1 describes the authenticated encryption called General Profile[3].

3.1.1 Security Analysis

Some supported instantiations include DES as the blockcipher, and MD4 and MD5 as the hash function. These are not good choices for known reasons. DES is an outdated standard, since its key and block sizes are too small considering modern computing power, and collisions have been found in MD4 and MD5 [87]. However, what our results show is that using the "more secure" building blocks, such as AES and a collision-resistant hash function, will not necessarily solve the problem. More precisely, we can neither prove, nor disprove the security of the General profile in this case. What we can show is that the Encode-then-Checksum-then-Encrypt composition method does not provide integrity in general when it uses a hash function as a checksum, even if it uses a secure encryption option for the underlying encryption scheme. We note that the theorem below does not imply that the General Profile (Construction 3.1.2) is insecure, but it shows the limitation of its underlying general design (Construction 3.1.1), when used with the given encoding scheme.

Theorem 3.1.3. Let $\mathcal{EC} = (Encode, Decode)$ be the encoding scheme, defined in Construction 3.1.2. There exists an IND-CPA secure encryption scheme, and a collision-resistant hash function, so that the authenticated encryption obtained via Encode-then-Checksum-then-Encrypt (Construction 3.1.1), does not provide integrity (is not INT-CTXT secure). Concretely, there exists an adversary I with reasonable resources, such that $\mathbf{Adv}^{\text{int-ctxt}}_{\mathcal{SE}'}(I) = 1$.

[3]Our analysis does not take into account stateful approaches for key derivation used in few options of General profile.

We present two alternative proofs of insecurity of the Encode-then-Checksum-then-Encrypt paradigm used in General Profile authenticated encryption. The first proof uses a natural encryption scheme and a not-so-natural hash function as a checksum. The second proof uses a special encryption scheme, but the checksum can be instantiated with arbitrary secure MAC. The attacks we provide are similar to those in [17, 66] that show insecurity of several general composition methods. We repeat that the attacks that we provide in the proof do not translate into an attack on any of the recommended options, because we use a special type of hash function/encryption scheme. They still show limitations in the general composition method.

First Proof of Theorem 3.1.3. Before presenting the proof, we give a high level idea. We show that the general authenticated encryption paradigm underlying General Profile does not preserve integrity of ciphertexts when instantiated with stateful counter (CTR) mode of the encryption scheme and a collision-resistant hash function that happens to leak the first bit of its input. The CTR mode of encryption is somewhat similar to the one-time pad, where the underlying blockcipher is applied to a counter to generate a pseudorandom pad which is then xor'ed with the message. Now, the ingenuity of our attack lies in showing that given any ciphertext that was output by the above scheme, one can produce another valid ciphertext by simply flipping the bits at two different positions, namely the first bit of the first and second blocks of the ciphertext. We repeat that the attack does not apply to the General Profile scheme itself.

Let $E\colon\ \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n$ be a blockcipher, and let $\mathcal{SE} = (\mathcal{K}_e, \mathcal{E}, \mathcal{D})$ be the associated stateful counter encryption scheme, known as CTR or xor encryption mode (cf. [12]). Its key generation algorithm $\mathcal{K}_e$ simply returns a random k-bit string K_e. The encryption algorithm $\mathcal{E}$ is stateful and maintains a counter ctr that is initially 0. $\mathcal{E}$ takes the key K_e, the current counter ctr, and a message M (padded if necessary to a length multiple of n bits), and outputs $ctr \,\|\, C[1] \,\|\, C[2] \,\|\, \ldots \,\|\, C[m]$, where m is the total number of blocks, and for $1 \leq i \leq m$, $C[i] \leftarrow M_i \oplus E_{K_e}(\langle ctr + i \rangle)$. Here $\langle i \rangle$ denotes the n-bit representation of

an integer i. Next, $\mathcal{E}$ updates the counter to $ctr + m + 1$. The decryption algorithm $\mathcal{D}$ takes K_e and a ciphertext $ctr \| C[1] \| ... \| C[m]$, and outputs $M[1] \| ... \| M[m]$, where for $1 \leq i \leq m$, $M[i] \leftarrow C[i] \oplus E_{K_e}(\langle ctr + i \rangle)$. The CTR encryption mode is proven to be IND-CPA secure if E is a PRF [12].

Let $\mathcal{H}$ be a collision-resistant hash function family, where each $h \in \mathcal{H}$ is a mapping from $\{0,1\}^*$ to $\{0,1\}^l$. Consider a modified hash function family $\mathcal{H}'$, i.e. every $h \in \mathcal{H}$ is modified to another $h' \in \mathcal{H}'$ as follows: for any message M, $h'(M) = M_{[0]} \| h(M_{[1...|M|-1]})$. We show that $\mathcal{H}'$ is also collision-resistant.

For any adversary A that can find collisions in $\mathcal{H}'$, we construct an adversary B that can find collisions in $\mathcal{H}$. When A returns two messages M', N', B computes $M \leftarrow M'_{[1...|M'|-1]}$, $N \leftarrow N'_{[1...|N'|-1]}$, and outputs M, N. Note that $h'(M') = M'_{[0]} || h\left(M'_{[1...|M'|-1]}\right) = M'_{[0]} || h(M)$, and $h(N') = N'_{[0]} || h\left(N'_{[1...|N'|-1]}\right) = N'_{[0]} || h(N)$.

If $h'(M') = h'(N')$, and $M' \neq N'$, then it is easy to see that $h(M) = h(N)$, and $M \neq N$. B is almost as efficient as A. Therefore, if $\mathcal{H}$ is collision resistant, then so is $\mathcal{H}'$.

We now present an adversary I that breaks the INT-CTXT security of the scheme described by Construction 3.1.1 when it uses CTR encryption mode and modified hash function family $\mathcal{H}'$ as $\mathcal{SE}$ and $\mathcal{CS}$, respectively. I selects an arbitrary n-bit-long message M and queries it to the encryption oracle. Let $ctr||C$ be the oracle's reply. I then queries the ciphertext $ctr||C'$ to the verification oracle, where C' is computed from C by flipping the first bit of the first and second blocks.

We claim that the int-ctxt advantage of I is 1. This is justified as follows. Consider $conf||\sigma||M = \mathcal{D}_{K_e}(ctr||C)$. Here, $\sigma = h'(M_t)$, and $M_t = conf||0^{l+1}||M$. So $\sigma = conf_{[0]}||h(M_{t[1...|M_t|-1]})$.

$ctr||C$ can be parsed as $ctr||C[1]||C[2]||D$, where $C[1]$ and $C[2]$ are the first and second blocks of C, and D is the remaining part of C. From the description of CTR encryption mode it is clear that

$C[1] = conf \oplus E_{K_e}(\langle ctr + 1 \rangle)$, and

$C[2] = (\sigma || M_{[0...n-l-2]}) \oplus E_{K_e}(\langle ctr+2 \rangle)$, where $\sigma = conf_{[0]} || h\left(M_{t[1...|M_t|-1]}\right)$, and $M_{[0...n-l-2]}$ is the first $n-(l+1)$ bits of M.

So $C[2] = \left(conf_{[0]} || h\left(M_{t[1...|M_t|-1]}\right) || M_{[0...n-l-2]}\right) \oplus E_{K_e}(\langle ctr+2 \rangle)$.

Let us denote the ciphertext blocks produced by flipping the first bit of $C[1]$ and $C[2]$ by $C'[1]$ and $C'[2]$, respectively. So we have

$C'[1] = \left(\overline{conf_{[0]}} || conf_{[1...n-1]}\right) \oplus E_{K_e}(\langle ctr+1 \rangle)$ and

$C'[2] = \left(\overline{conf_{[0]}} || \mathcal{H}_{K_h}\left(M_{t[1...|M_t|-1]}\right) || M_{[0...n-l-2]}\right) \oplus E_{K_e}(\langle ctr+2 \rangle)$.

Let us denote the decryption of $ctr||C'$ by $(M'_{el}||\sigma'||M'_{er})$. So we have $M'_{el} = (\overline{conf_{[0]}}||conf_{[1...n-1]})$, $\sigma' = (\overline{conf_{[0]}}||h(M_{t[1...|M_t|-1]}))$, and $M'_{er} = M$.

Now notice that

$M'_e = (M'_{el}||M'_{er}) = (\overline{conf_{[0]}}||conf_{[1...n-1]})||M$, and

$M'_t = (M'_{el}||0^{l+1}||M'_{er}) = (\overline{conf_{[0]}}||conf_{[1...n-1]})||0^{l+1}||M$.

It is clear $M_{t[1...|M_t|-1]} = M'_{t[1...|M'_t|-1]}$ because M_t and M'_t differ only in first bit. So from above, we have

$\sigma' = \overline{conf_{[0]}}||h(M_{t[1...|M_t|-1]}) = \overline{conf_{[0]}}||h(M'_{t[1...|M'_t|-1]}) = h'(M'_t)$.

Thus, (M'_t, σ') is a valid message-tag pair. Hence $ctr||C'$ is a valid ciphertext that was never returned by the encryption oracle, and therefore, the int-ctxt advantage of I is 1.

I makes one oracle query of length n bits, and performs two operations of bit complementation. □

Second Proof of Theorem 3.1.3. Our second proof is relatively simpler than the first one. We show that the general authenticated encryption paradigm underlying General Profile does not preserve integrity of ciphertexts when instantiated with any arbitrary encoding scheme, an unforgeable MAC, and a special type of IND-CPA secure encryption scheme whose encryption algorithm prepends zero to the ciphertext and the decryption algorithm simply ignores the first bit of the ciphertext. In the Encode-then-Checksum-then-Encrypt paradigm, encryption is the last step. So, with the above mentioned special type of encryption, one can easily produce a new valid ciphertext C', given any ciphertext C output by the

above scheme by flipping the first bit of C. We repeat that the attack does not apply to the General Profile scheme itself.

Let $\mathcal{SE} = (\mathcal{K}_e, \mathcal{E}, \mathcal{D})$ be any IND-CPA secure encryption scheme. Consider a modified encryption scheme $\mathcal{SE}'' = (\mathcal{K}_e, \mathcal{E}'', \mathcal{D}'')$, where $\mathcal{E}''$ on input key K_e and a message M outputs $0||\mathcal{E}_{K_e}(M)$, and $\mathcal{D}''$ on input key K_e and a ciphertext C outputs $\mathcal{D}_{K_e}(C_{[1...|C|-1]})$. It is easy to see that if $\mathcal{SE}$ is IND-CPA secure, then so is $\mathcal{SE}''$ (cf. [17], Proof of IND-CPA security of $\mathcal{SE}_2$, at the end of Section 3). Let $\mathcal{MAC} = (\mathcal{K}_t, \mathcal{T})$ be any UF-CMA secure MAC.

We present an adversary I attacking INT-CTXT security of the scheme described by Construction 3.1.1 when it uses $\mathcal{SE}''$ and $\mathcal{MAC}$ as the encryption and checksum component schemes. Note that we did not make any assumption about the encoding scheme, so this attack works for any arbitrary encoding scheme. I selects an arbitrary short message M in the message space of the scheme. It queries this message to the encryption oracle and gets back ciphertext C. I then flips the first bit of C and queries the resulting ciphertext $C' = 1||C_{[1...|C|-1]}$ to the verification oracle.

It is clear that $C' \neq C$, and C' is a valid ciphertext, because $\mathcal{D}''$ ignores the first bit of ciphertext; therefore, $\mathcal{D}''_{K_e}(C') = \mathcal{D}''_{K_e}(C) = M$. Thus, the int-ctxt advantage of I is 1. I makes only one oracle query of length $|M|$, and performs one bit complementation. □

3.1.2 Modified General Profile

We now suggest simple and easy-to-implement modifications to the General profile construction, and show that they are sufficient to prove the security of the scheme. Namely, we suggest to use a secure MAC in place of the hash function, and show that the resulting authenticated encryption scheme is secure. Note that this does not contradict Theorem 3.1.3, because now we look at the particular encryption scheme that the General profile uses (Construction 3.1.2), i.e., CBC with zero IV. We now define the construction, and state its security.

Construction 3.1.4. [Modified General profile] The construction is like Construction 3.1.2, except that a message authentication code $\mathcal{MAC} = (\mathcal{K}_t, \mathcal{T})$ is used as checksum $\mathcal{CS}$.

Theorem 3.1.5. The authenticated encryption scheme described by the Modified General Profile (Construction 3.1.4) is INT-CTXT and IND-CCA secure, if the underlying blockcipher is a PRF, and the underlying checksum (MAC) is a PRF.

Concretely, let $E\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher, and $\mathcal{SE} = (\mathcal{K}_e, \mathcal{E}, \mathcal{D})$ be the CBC encryption mode with $IV = 0^n$, that uses E. Let $\mathcal{MAC} = (\mathcal{K}_t, \mathcal{T})$ be a message authentication code with output of length l. Let $\mathcal{EC} = (Encode, Decode)$ be an encoding scheme, and $\mathcal{SE}' = (\mathcal{K}', \mathcal{E}', \mathcal{D}')$ be the authenticated encryption scheme associated to them by Modified General Profile (Construction 3.1.4). Then, for any adversary I attacking the INT-CTXT security of $\mathcal{SE}'$, that runs in time t, and makes q'_e queries to the encryption oracle, and q_v queries to the verification oracle, totaling μ'_e and μ_v bits, respectively, there exists an adversary F attacking the UF-CMA security of $\mathcal{MAC}$, such that

$$\mathbf{Adv}^{\text{int-ctxt}}_{\mathcal{SE}'}(I) \;\leq\; \mathbf{Adv}^{\text{uf-cma}}_{\mathcal{MAC}}(F)\,. \tag{1}$$

Furthermore, F runs in time t, and makes q'_e queries to the tagging oracle,and q_v queries to the verification oracle, totaling at most $\mu'_e + q'_e \cdot (2n + l - 1)$ and $\mu_v + q_v \cdot (2n + l - 1)$ bits, respectively.

And for any adversary A attacking the IND-CCA security of $\mathcal{SE}'$, that runs in time t, and makes q_e queries to the left-right encryption oracle and q_d queries to the decryption oracle, totaling μ_e and μ_d bits, respectively, there exist adversaries B and G attacking PRF security of E and $\mathcal{MAC}$, respectively, such that

$$\mathbf{Adv}^{\text{ind-cca}}_{\mathcal{SE}'}(A) \;\leq\; \mathbf{Adv}^{\text{prf}}_{E}(B) + 4 \cdot \mathbf{Adv}^{\text{prf}}_{\mathcal{MAC}}(G) + \frac{\mu_e^2}{n^2 \cdot 2^n} + \frac{2 \cdot q_d}{2^l}\,. \tag{2}$$

Furthermore, B runs in time t, and makes at most $\lfloor(\mu_e + q_e \cdot (2n + l - 1)/n)\rfloor$ oracle queries, totaling at most $\mu_e + q_e \cdot (2n + l - 1)$ bits; G runs in time t, and makes $q_e + q_d$ oracle queries, totaling at most $\mu_e + \mu_d + (q_e + q_d)(2n + l - 1)$ bits.

Next we present the proof. Note that the INT-CTXT security of the scheme requires only UF-CMA security of the checksum (MAC), while IND-CCA security relies on it being a PRF. As we mentioned before, any PRF MAC is also UF-CMA (Theorem 2.3.5), so PRF security is a sufficient assumption.

AES is believed to be a PRF, and HMAC was shown to be a PRF [10], assuming the underlying compression function is a PRF (cf. [10] for the definition of the latter). Therefore, these schemes constitute good instantiations for the above design.

Proof of Theorem 3.1.5. INT-CTXT SECURITY. We will reduce the integrity of ciphertexts of the Modified General Profile to the unforgeability of the underlying MAC scheme. The attack in First Proof of Theorem 3.1.3 shows that a collision-resistant hash function is not sufficient for integrity of ciphertexts. Moreover, from the attack in Second Proof of Theorem 3.1.3, we know that even an unforgeable MAC cannot provide integrity of ciphertexts if used with any general IND-CPA secure encryption scheme. At a high level, we need an unforgeable MAC, and the encryption scheme is required to have the following property for integrity of ciphertexts: for any pair of ciphertexts c, c', if $c \neq c'$ then $m \neq m'$, where m, m' are the corresponding plaintexts. It is easy to see that while CBC with zero IV mode of encryption (or, any other standard deterministic encryption mode) satisfies this property, it may not necessarily hold for any general IND-CPA secure encryption scheme.

We now justify Eq. (1). Let I be an adversary attacking the INT-CTXT security of $\mathcal{SE}'$. We construct a forger F breaking the UF-CMA security of $\mathcal{MAC}$. F first runs $\mathcal{K}_e$ to obtain a key K_e for $\mathcal{E}$. It runs I and replies to its queries as follows.

For every encryption oracle query M that I makes, F does the following: It computes $(M_e, M_t) \xleftarrow{\$} Encode(M)$, and then it queries M_t to its own tagging oracle. Let us call the oracle's reply σ. Next, F parses M_e as $M_{el}||M_{er}$, and forms $M_{el}||\sigma||M_{er}$. Then, it computes $C \leftarrow \mathcal{E}_{K_e}(M_{el}||\sigma||M_{er})$ and returns C to I.

For every verification oracle query C that I makes, F does the following: It computes

$M_{el}||\sigma||M_{er} \leftarrow \mathcal{D}_{K_e}(C)$ and $M_t \leftarrow Decode(M_e)$, where $M_e = M_{el}||M_{er}$. Next, F queries (M_t, σ) to its own verification oracle, then returns 1 to I if the same was returned by its own oracle.

We now analyze F. We claim that F is successful whenever I is successful. First of all, it is straightforward to see that F correctly simulates the encryption oracle for I. Now, if I is successful, then one of its verification oracle queries C' is such that it was not returned by the encryption oracle (i.e. it's new), and its decryption does not return $\perp$. This means that M'_e must be new, where $M'_e(= M'_{el}||\sigma'||M'_{er}) \leftarrow \mathcal{D}_{K_e}(C')$, because the base encryption scheme $\mathcal{SE}$ is deterministic (CBC with zero IV). If $(M'_{el}||\sigma'||M'_{er})$ is new, then either $M'_{el}||M'_{er}$ or σ' must be new, which is equivalent to saying that either $M'_{el}||0^n||M'_{er}(= M'_t)$ or σ' must be new. This gives rise to two cases. The first case is when M'_t is new (σ' may or may not be new in this case). It is clear that in this case (M'_t, σ') is a valid new message-tag pair. Hence F's verification oracle will return 1.

The second case is when only σ' is new, and M'_t is old, i.e. M'_t is one of the messages that was queried to the tagging oracle. But then in this case, σ' is an invalid tag, as the tagging algorithm is deterministic, and the distinct valid tag was returned as the answer to the corresponding query to the tagging oracle. Hence decryption of C' will return $\perp$.

Hence, the uf-cma advantage of F is the same as the int-ctxt advantage of I. The time complexity of F is basically that of I. F makes the same number of oracle queries as that of I. The total length of all the queries made by F exceeds that of I by only a fixed number of bits, which is the number of queries times $(2n + l - 1)$, due to the use of encoding (at most $n - 1$ bits for padding, n bits for confounder, and l bits for tag).

We now claim the IND-CPA security of $\mathcal{SE}'$. IND-CCA security will then follow from the IND-CPA security and INT-CTXT security of the scheme.

IND-CPA SECURITY. We show that the composed encryption scheme $\mathcal{SE}'$ is IND-CPA secure if the underlying blockcipher is a PRF and the underlying MAC is a PRF.

Lemma 3.1.6. For any adversary S attacking IND-CPA security of $\mathcal{SE}'$, that runs in time

t, and makes q queries to the left-right encryption oracle, totaling μ bits, there exist adversaries B and G attacking PRF security of E and $\mathcal{MAC}$, such that

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}'}(S) \leq \mathbf{Adv}^{\text{prf}}_{E}(B) + 2 \cdot \mathbf{Adv}^{\text{prf}}_{\mathcal{MAC}}(G) + \frac{\mu^2}{n^2 \cdot 2^n} \ .$$

Furthermore, B runs in time t and makes at most $\lfloor \mu + q \cdot (2n + l - 1)/n) \rfloor$ oracle queries, totaling at most $\mu + q \cdot (2n + l - 1)$ bits; G runs in time t and makes q oracle queries, totaling at most $\mu + q \cdot (2n + l - 1)$ bits.

Proof of Lemma 3.1.6. We will first show that if the underlying MAC is PRF, then the encryption in the Modified General Profile is similar in terms of security to CBC encryption with random IV (Claim 3.1.7). Next, from the well known result of [20] (Claim 3.1.8), we know that CBC encryption with random IV is IND-CPA secure if the underlying blockcipher is a PRF. So, Lemma 3.1.6 will follow immediately from Claim 3.1.7 and Claim 3.1.8.

□

Claim 3.1.7. For any adversary S attacking IND-CPA security of $\mathcal{SE}'$, that runs in time t, and makes q queries to the left-right encryption oracle, totaling μ bits, there exists an adversary D attacking IND-CPA security of CBC encryption scheme with random IV CBC$ = $(\mathcal{K}_e, \mathcal{E}^{\$}, \mathcal{D}^{\$})$, and an adversary G attacking PRF security of $\mathcal{MAC}$, such that

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}'}(S) \leq \mathbf{Adv}^{\text{ind-cpa}}_{CBC\$}(D) + 2 \cdot \mathbf{Adv}^{\text{prf}}_{\mathcal{MAC}}(G) \ .$$

Furthermore, D runs in time t and makes q queries to the left-right encryption oracle, totaling at most $(\mu + q \cdot (2n + l - 1))$ bits; G runs in time t and makes q oracle queries, totaling at most $(\mu + q \cdot (2n + l - 1))$ bits.

We recall a fact from [20].

Claim 3.1.8. **[[20], Theorem 4.19]** Let $E\colon\ \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher, and let CBC$ = $(\mathcal{K}_e, \mathcal{E}^{\$}, \mathcal{D}^{\$})$ be the associated CBC encryption scheme with random IV (cf. [12]). Then for any adversary D attacking IND-CPA security of CBC$, that runs in time t

and makes q queries to the left-right encryption oracle, totaling μ n-bit blocks, there exists an adversary B attacking PRF security of E, such that

$$\mathbf{Adv}^{\text{ind-cpa}}_{CBC\$}(D) \leq \mathbf{Adv}^{\text{prf}}_{E}(B) + \frac{\mu^2}{2^n} \ .$$

Furthermore, B runs in time[4] t and makes μ oracle queries, totaling μn bits.

Proof of Claim 3.1.7. At a high level the proof follows from the observation that the encoding scheme of Modified General Profile prepends a random confounder to the plaintext. So encrypting this encoded message using CBC with zero IV is equivalent to encrypting any message using CBC with "pseudorandom" IV, because the underlying blockcipher is assumed to be a PRF.

Let S be an adversary attacking IND-CPA security of $\mathcal{SE}'$. For $x \in \{0, 1, 2, 3, 4, 5\}$, we define the following experiments associated with S.

Experiment $\mathbf{ExpH}_x$

$K_e \xleftarrow{\$} \mathcal{K}_e,\ K_t \xleftarrow{\$} \mathcal{K}_t$

Run S replying to its oracle query (M, N) as follows:

$(M_e, M_t) \xleftarrow{\$} Encode(M)$; $(N_e, N_t) \xleftarrow{\$} Encode(N)$; $r \xleftarrow{\$} \{0,1\}^n$

Parse M_e and N_e as $M_{el}||M_{er}$ and $N_{el}||N_{er}$, where $|M_{el}| = |N_{el}| = n$

Switch (x):

Case $x = 0$: $\sigma \leftarrow \mathcal{T}_{K_t}(M_t)$; $C \leftarrow \mathcal{E}_{K_e}(M_{el}||\sigma||M_{er})$

Case $x = 1$: $C \leftarrow \mathcal{E}_{K_e}(M_{el}||r||M_{er})$

Case $x = 2$: $IV||C \xleftarrow{\$} \mathcal{E}^{\$}_{K_e}(M_{el}||r||M_{er})$

Case $x = 3$: $IV||C \xleftarrow{\$} \mathcal{E}^{\$}_{K_e}(N_{el}||r||N_{er})$

Case $x = 4$: $C \leftarrow \mathcal{E}_{K_e}(N_{el}||r||N_{er})$

Case $x = 5$: $\sigma \leftarrow \mathcal{T}_{K_t}(N_t)$; $C \leftarrow \mathcal{E}_{K_e}(N_{el}||\sigma||N_{er})$

Return C to S.

When S halts and outputs a bit, return that bit.

[4]Due to the difference in convention, this time complexity is different from the one given in [20].

For $x \in \{0, 1, 2, 3, 4, 5\}$, let $P_x = \Pr[\ \mathbf{ExpH}_x = 1\ ]$ denote the probability that $\mathbf{ExpH}_x$ returns 1. By the definition of $\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}'}(S)$, we have

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}'}(S) = P_5 - P_0 = (P_5 - P_4) + (P_4 - P_3) + (P_3 - P_2) + (P_2 - P_1) + (P_1 - P_0)\ . \quad (3)$$

We first show that for S, $\mathbf{ExpH}_1$ is indistinguishable from $\mathbf{ExpH}_2$. In $\mathbf{ExpH}_1$, $(M_{el}||r||M_{er})$ is encrypted using the CBC mode with zero IV, and the whole ciphertext is returned to S, while in $\mathbf{ExpH}_2$, $(M_{el}||r||M_{er})$ is encrypted using the CBC mode with random IV, and the whole ciphertext, except the IV, is returned to S. Note that in the latter case, the ciphertext given to S has the form of $((M_{el} \oplus IV)||r||M_{er})$ encrypted using the CBC mode with zero IV. However, since M_{el} and IV are uniformly random strings, to an adversary that doesn't know these in advance, M_{el} and $(M_{el} \oplus IV)$ are indistinguishable. Hence, adversary S cannot distinguish between $\mathbf{ExpH}_1$ and $\mathbf{ExpH}_2$. The same argument applies to show that experiments $\mathbf{ExpH}_3$ and $\mathbf{ExpH}_4$ are indistinguishable in S's view. Hence, $(P_4 - P_3) = (P_2 - P_1) = 0$. Thus, we have

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}'}(S) = (P_5 - P_4) + (P_3 - P_2) + (P_1 - P_0)\ . \quad (4)$$

Given S, there exist adversaries D and G, such that the following claims hold, and these adversaries use the resources specified in Claim 3.1.7.

Claim 3.1.9. $P_3 - P_2 \leq \mathbf{Adv}^{\text{ind-cpa}}_{CBC\$}(D)$.

Claim 3.1.10. $(P_5 - P_4) + (P_1 - P_0) \leq 2 \cdot \mathbf{Adv}^{\text{prf}}_{\mathcal{MAC}}(G)$.

Eq. (4) and the above claims imply Claim 3.1.7. □

Proof of Claim 3.1.9. We construct an adversary D breaking the IND-CPA security of CBC\$ using adversary S as follows. For every message-pair query (M, N) that S makes, D first computes $(M_e, M_t) \xleftarrow{\$} Encode(M)$, $(N_e, N_t) \xleftarrow{\$} Encode(N)$, $r \xleftarrow{\$} \{0,1\}^l$. Next, it parses M_e and N_e as $M_{el}||M_{er}$ and $N_{el}||N_{er}$, and it queries $(M_{el}||r||M_{er}, N_{el}||r||N_{er})$ to its own oracle to get back $IV||C$, where IV is the first ciphertext block. D forwards C back to S. When S halts and returns a bit, D halts and outputs that bit.

We analyze D. The view of S in $\mathbf{ExpH}_2$ is indistinguishable from that in $\mathbf{Exp}^{\text{ind-cpa-0}}_{CBC\$}(D)$, and the view of S in $\mathbf{ExpH}_3$ is indistinguishable from that in $\mathbf{Exp}^{\text{ind-cpa-1}}_{CBC\$}(D)$. Thus, $P_3 - P_2 \leq \mathbf{Adv}^{\text{ind-cpa}}_{CBC\$}(D)$.

The time complexity of D is basically that of S. D makes the same number of oracle queries as S. The total length of all the queries made by D exceeds that of S by only a fixed number of bits, which is the number of queries times $(2n + l - 1)$, due to the use of encoding (at most $(n - 1)$ bits for padding, n bits for confounder, and l bits for tag). □

Proof of Claim 3.1.10. We construct adversaries G_1 and G_2 breaking the PRF security of $\mathcal{MAC}$ using adversary S such that

$$(P_5 - P_4) + (P_1 - P_0) \leq \mathbf{Adv}^{\text{prf}}_{\mathcal{MAC}}(G_2) + \mathbf{Adv}^{\text{prf}}_{\mathcal{MAC}}(G_1) \ .$$

G_1 runs $\mathcal{K}_e$ to obtain a key K_e. For every message-pair query (M, N) that S makes, G_1 first computes $(M_e, M_t) \xleftarrow{\$} \mathit{Encode}(M)$. Then it queries M_t to its oracle. Let's call the oracle's reply σ. Next, it parses M_e as $M_{el}||M_{er}$, forms $M_{el}||\sigma||M_{er}$, and computes $C \leftarrow \mathcal{E}_{K_e}(M_{el}||\sigma||M_{er})$. G_1 forwards C back to S. When S halts and returns a bit, G_1 halts and outputs the complement bit.

We analyze G_1. When G_1 is in the first experiment of Definition 2.3.4, then $\sigma = \mathcal{T}_{K_t}(M_t)$, so G_1 simulates $\mathbf{ExpH}_0$ perfectly, and when G_1 is in the second experiment of Definition 2.3.4, then σ is a random n-bit string, so G_1 simulates experiment $\mathbf{ExpH}_1$ perfectly. Hence, $(P_1 - P_0) \leq \mathbf{Adv}^{\text{prf}}_{\mathcal{MAC}}(G_1)$.

Adversary G_2 can be constructed in a similar way, where for every message-pair query (M, N), it does similar things as G_1, but for message N. Thus, we have $(P_5 - P_4) \leq \mathbf{Adv}^{\text{prf}}_{\mathcal{MAC}}(G_2)$.

The time complexities of G_1, G_2 are basically that of S. G_1, G_2 make the same number of oracle queries as that of S. The total length of all the queries made by G_1, G_2 exceed that of S by only a fixed number of bits, which is number of queries times $(2n + l - 1)$, due to the use of encoding (at most $(n - 1)$ bits for padding, n bits for confounder, and l bits for

tag).

Putting G to be one of the adversaries G_1, G_2 with the larger prf-advantage we get

$$\mathbf{Adv}^{\text{prf}}_{\mathcal{MAC}}(G_2) + \mathbf{Adv}^{\text{prf}}_{\mathcal{MAC}}(G_1) \leq 2 \cdot \mathbf{Adv}^{\text{prf}}_{\mathcal{MAC}}(G) \ .$$

Thus, Claim 3.1.10 follows. □

IND-CCA SECURITY. Eq. (1), Lemma 3.1.6, Theorem 2.2.4, and Theorem 2.3.5 imply Eq. (2).

□

3.2 Simplified Profile

Kerberos designers proposed a new construction that they call "Simplified Profile" (cf. Section 5 in [82], and [81]). Again, we start with a more general composition method that outlines the design.

Construction 3.2.1. [Encode-then-Encrypt&MAC] Let $\mathcal{SE} = (\mathcal{K}_e, \mathcal{E}, \mathcal{D})$, $\mathcal{MAC} = (\mathcal{K}_t, \mathcal{T})$, and $\mathcal{EC} = (\mathit{Encode}, \mathit{Decode})$ be an encryption scheme, a MAC scheme, and an encoding scheme. The message space of corresponding *Encode-then-Encrypt&MAC* scheme $\mathcal{SE}' = (\mathcal{K}', \mathcal{E}', \mathcal{D}')$, is that of $\mathcal{EC}$, and the algorithms are defined as follows:

- $\mathcal{K}'$ runs $\mathcal{K}_e, \mathcal{K}_t$, and returns their outputs $K_e \parallel K_t$.
- $\mathcal{E}'$ on inputs $K_e \parallel K_t$ and M, first gets the encodings via $(M_e, M_t) \xleftarrow{\$} \mathit{Encode}(M)$. It then computes $C \leftarrow \mathcal{E}_{K_e}(M_e)$, $\sigma \leftarrow \mathcal{T}_{K_t}(M_t)$, and returns $C \parallel \sigma$.
- $\mathcal{D}'$ on inputs $K_e \parallel K_t$ and $C \parallel \sigma$, computes $M_e \leftarrow \mathcal{D}_{K_e}(C)$, decodes $(M, M_t) \leftarrow \mathit{Decode}(M_e)$, computes $\sigma' \leftarrow \mathcal{T}_{K_t}(M_t)$, and returns M, if $\sigma = \sigma'$, and $\perp$ otherwise.

Above we assume that the outputs of the encoding scheme are compatible with the inputs to $\mathcal{E}, \mathcal{T}$.

The next construction defines the Simplified profile, and Figure 3 depicts the design.

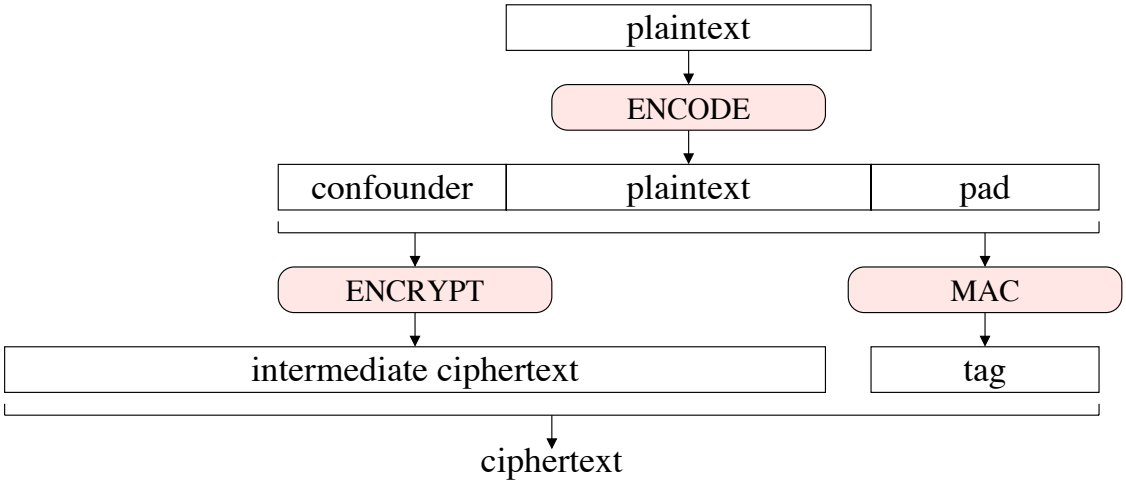

Figure 3: Encode-then-Encrypt&MAC Paradigm

Construction 3.2.2. [Authenticated encryption in Kerberos: Simplified profile] Let $E\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher. Let $\mathcal{SE} = (\mathcal{K}_e, \mathcal{E}, \mathcal{D})$ be the associated CBC encryption mode with $IV = 0^n$. Let $\mathcal{MAC} = (\mathcal{K}_t, \mathcal{T})$ be a MAC scheme. Let $\mathcal{EC} = (Encode, Decode)$ be an encoding scheme, such that $Encode$ with $\mathsf{MsgSp} = \{0,1\}^*$ on input M pads M to make its length a multiple of n bits (while permitting unambiguous decoding), picks a random confounder of n bits $conf \xleftarrow{\$} \{0,1\}^n$, computes $M_e \leftarrow conf \,\|\, M$, and $M_t \leftarrow conf \,\|\, M$, and returns (M_e, M_t). $Decode$ on input M_e, parses it as $conf \| M$, computes $M_t \leftarrow M_e$, and returns (M, M_t). Then Construction 3.2.1 describes the Simplified Profile of authenticated encryption in Kerberos.

3.2.1 Security Analysis

The following theorem states that the Simplified Profile provides strong security guarantees.

Theorem 3.2.3. The authenticated encryption scheme $\mathcal{SE}'$, described by the Simplified Profile (Construction 3.2.2), is INT-CTXT and IND-CCA secure, if the underlying blockcipher E is a PRF and the underlying MAC is a PRF.

Concretely, let $E\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher, and $\mathcal{SE} = (\mathcal{K}_e, \mathcal{E}, \mathcal{D})$ be the CBC encryption mode with $IV = 0^n$ that uses E. Let $\mathcal{MAC} = (\mathcal{K}_m, \mathcal{T})$ be a MAC scheme

and $\mathcal{EC} = (Encode, Decode)$ be an encoding scheme. Let $\mathcal{SE}'$ be the authenticated encryption scheme associated to them by Simplified Profile (Construction 3.2.2). Then, for any adversary I attacking INT-CTXT security of $\mathcal{SE}'$, that runs in time t, and makes q'_e queries to the encryption oracle, and q_v queries to the verification oracle, totaling μ'_e and μ_v bits, respectively, there exists an adversary F attacking UF-CMA security of $\mathcal{MAC}$, such that

$$\mathbf{Adv}^{\text{int-ctxt}}_{\mathcal{SE}'}(I) \;\leq\; \mathbf{Adv}^{\text{uf-cma}}_{\mathcal{MAC}}(F)\,. \tag{5}$$

Furthermore, F runs in time t and makes q'_e queries to the tagging oracle and q_v queries to the verification oracle, totaling at most $\mu'_e + q'_e \cdot (2n-1)$ and $\mu_v + q_v \cdot (2n-1)$ bits, respectively. And for any adversary A attacking IND-CCA security of $\mathcal{SE}'$, that runs in time t and makes q_e queries to the left-right encryption oracle, and q_d queries to the decryption oracle, totaling μ_e and μ_d bits, respectively, there exist adversaries B and G attacking PRF security of E and $\mathcal{MAC}$, respectively, such that

$$\mathbf{Adv}^{\text{ind-cca}}_{\mathcal{SE}'}(A) \;\leq\; \mathbf{Adv}^{\text{prf}}_{E}(B) + 4 \cdot \mathbf{Adv}^{\text{prf}}_{\mathcal{MAC}}(G) + \frac{q_e(q_e-1)}{2^{n+1}} + \frac{\mu_e^2}{n^2 \cdot 2^n} + \frac{2 \cdot q_d}{|\mathsf{Ran}_{\mathcal{T}}|}\,, \tag{6}$$

where $\mathsf{Ran}_{\mathcal{T}}$ denotes the set of outputs of $\mathcal{T}$. Furthermore, B runs in time t and makes at most $\lfloor (\mu_e + q_e \cdot (2n-1)/n) \rfloor$ oracle queries, totaling at most $\mu_e + q_e \cdot (2n-1)$ bits; G runs in time t and makes $q_e + q_d$ oracle queries, totaling at most $\mu_e + \mu_d + (q_e + q_d) \cdot (2n-1)$ bits.

Next we present the proof. Note that INT-CTXT security of the scheme requires only UF-CMA security of the MAC, while IND-CCA security relies on the MAC being a PRF. As we mentioned before, any PRF MAC is also UF-CMA (Theorem 2.3.5), so PRF security is a sufficient assumption. Also, AES is believed to be a PRF, and HMAC was shown to be a PRF [10], assuming the underlying compression function is a PRF (cf. [10] for the definition of the latter notion). Therefore, these schemes constitute good instantiations for the above design.

Proof of Theorem 3.2.3. INT-CTXT SECURITY. We will reduce the integrity of ciphertexts of Simplified Profile to the unforgeability of the underlying MAC scheme. First, we note

that an attack similar to that in Second Proof of Theorem 3.1.3 can be mounted on Simplified Profile, too. Hence, as pointed out in Section 3.1.2, for the integrity of ciphertexts it is necessary that the encryption scheme satisfies the following property: for any pair of ciphertexts c, c', if $c \neq c'$ then $m \neq m'$, where m, m' are the corresponding plaintexts. In addition, we point out again that while CBC with zero IV mode of encryption (or, any other standard deterministic encryption mode) satisfies this property, it may not necessarily hold for any general IND-CPA secure encryption scheme.

We justify Eq. (5). Let I be an adversary attacking INT-CTXT security of $\mathcal{SE}'$. We construct a forger F breaking the UF-CMA security of $\mathcal{MAC}$. F first runs $\mathcal{K}_e$ to obtain a key K_e for $\mathcal{E}$. It runs I and replies to its queries as follows.

For every encryption oracle query M that I makes, F does the following: It computes $(M_e, M_t) \xleftarrow{\$} \mathit{Encode}(M)$ and then queries M_t to its own tagging oracle. Let us call the oracle's reply σ. Next, F computes $C \leftarrow \mathcal{E}_{K_e}(M_e)$ and returns $C||\sigma$ to I.

For every verification oracle query $C||\sigma$ that I makes, F does the following: It computes $M_e \leftarrow \mathcal{D}_{K_e}(C)$ and $M_t \leftarrow \mathit{Decode}(M_e)$. Next, F queries (M_t, σ) to its own verification oracle and returns 1 to I, if the same was returned by its own oracle.

We now analyze F. We claim that F is successful whenever I is successful. First of all, it is straightforward to see that F correctly simulates the encryption oracle for I. Now, if I is successful, then one of its verification oracle queries $C'||\sigma'$ is such that it was not returned by the encryption oracle (i.e. it's new), and its decryption does not return $\perp$. This gives rise to two cases. The first case is when C' is new (σ' may or may not be new in this case). In this case, $M'_e \leftarrow \mathcal{D}_{K_e}(C')$ must be new, because C' is new, and $\mathcal{SE}$ is deterministic. M'_t is new, because it is equal to M'_e. Thus, (M'_t, σ') is a valid new message-tag pair. Hence F's verification oracle will return 1.

The second case is when only σ' is new and C' is old. However, we show that in this case σ' is invalid, and therefore decryption of C' will return $\perp$. For the same reasons as explained above, old C' implies that M'_t is old, i.e. M'_t is one of the messages which was

queried to the tagging oracle. But then σ' is an invalid tag, as the corresponding valid and distinct tag was returned as the answer to the corresponding query.

Hence, the uf-cma advantage of F is the same as the int-ctxt advantage of I. The time complexity of F is basically that of I. F makes the same number of oracle queries as that of I. The total length of all the queries made by F exceeds that of I by only a fixed number of bits, which is the number of queries times $(2n - 1)$, due to the use of encoding (at most $(n - 1)$ bits for padding, and n bits for confounder).

Before we analyze the IND-CCA security of $\mathcal{SE}'$, let us claim its IND-CPA security.

IND-CPA SECURITY. Theorem 7.1 from [15] states that an encryption scheme composed via the Encode-then-Encrypt&MAC paradigm is IND-CPA if the base encoding scheme is Coll-CPA, the base MAC scheme is PRF, and the base encryption scheme is IND-CPA. However, we cannot use it directly, because the base encryption scheme in Construction 3.2.2 is CBC with fixed IV, which is obviously not IND-CPA. We present a modification of Theorem 7.1 from [15] to claim the IND-CPA security of the encryption scheme in Construction 3.2.2, but before that we present the following construction and its security analysis:

Construction 3.2.4. Let $\mathcal{SE} = (\mathcal{K}_e, \mathcal{E}, \mathcal{D})$ be the CBC encryption scheme with $IV = 0^n$, and $\mathcal{EC} = (Encode, Decode)$ be the encoding scheme of Construction 3.2.2. Then, $\mathcal{SE}'' = (\mathcal{K}_e, \mathcal{E}'', \mathcal{D}'')$ is defined as follows.

- $\mathcal{E}''$ on inputs K_e and M first gets the encodings via $(M_e, M_t) \xleftarrow{\$} Encode(M)$. It then computes $C \leftarrow \mathcal{E}_{K_e}(M_e)$, parses M_e as $conf||M$, where $|conf| = n$, and returns $conf||C$.

- $\mathcal{D}''$ on inputs K_e and $conf||C$ computes $M_e \leftarrow \mathcal{D}_{K_e}(C)$, decodes $(M, M_t) \leftarrow Decode(M_e)$, and returns M.

Claim 3.2.5. The scheme $\mathcal{SE}''$ defined in Construction 3.2.4 is as secure as the CBC encryption scheme with random IV, CBC\$ = $(\mathcal{K}_e, \mathcal{E}^\$, \mathcal{D}^\$)$. More precisely, for any adversary A attacking IND-CPA security of $\mathcal{SE}''$, that runs in time t, and makes q queries to the

left-right encryption oracle, totaling μ bits, there exists an adversary D attacking IND-CPA security of CBC\$, such that

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}''}(A) \leq \mathbf{Adv}^{\text{ind-cpa}}_{CBC\$}(D) \ .$$

Furthermore, D runs in time t and makes q queries to the left-right encryption oracle, totaling at most $(\mu + q \cdot (2n - 1))$ bits.

Proof of Claim 3.2.5. The proof follows from a similar observation (as in the proof of Claim 3.1.7) that the random confounder prepended to the message by the encoding scheme acts as a "pseudorandom" IV in the encryption, because the underlying blockcipher is assumed to be a PRF.

We construct an adversary D, breaking the IND-CPA security of CBC\$, using adversary A.

For every message-pair query (M, N) that A makes, D first computes $(M_e, M_t) \xleftarrow{\$} \mathit{Encode}(M)$, parses M_e as $\mathit{conf}||M$, where $|\mathit{conf}| = n$, pads N to multiple block lengths, and computes $N_e \leftarrow \mathit{conf}||N$. Then it queries (M_e, N_e) to its own oracle and gets back $IV||C$, where IV is the first ciphertext block. D forwards $(\mathit{conf} \oplus IV)||C$ back to A. When A halts and returns a bit, D halts and outputs that bit.

We analyze D. We claim that if D is in $\mathbf{Exp}^{\text{ind-cpa-b}}_{CBC\$}(D)$ for $b \in \{0, 1\}$, then A's view in the simulated experiment is the same as that in the actual experiment $\mathbf{Exp}^{\text{ind-cpa-b}}_{\mathcal{SE}''}(A)$. Hence $\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}''}(A) \leq \mathbf{Adv}^{\text{ind-cpa}}_{CBC\$}(D)$.

The time complexity of D is basically that of A. D makes the same number of oracle queries as that of A. The total length of all the queries made by D exceeds that of A by only a fixed number of bits, which is the number of queries times $(2n - 1)$, due to the use of encoding (at most $(n - 1)$ bits for padding, and n bits for confounder) □

From Claim 3.2.5 and Claim 3.1.8, we conclude the following.

Claim 3.2.6. The scheme $\mathcal{SE}''$ defined in Construction 3.2.4 is IND-CPA secure if the underlying blockcipher E is a PRF. More precisely, for any adversary A attacking IND-CPA security of $\mathcal{SE}''$, that runs in time t and makes q queries to the left-right encryption oracle, totaling μ bits, there exists an adversary B attacking PRF security of E, such that

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}''}(A) \leq \mathbf{Adv}^{\text{prf}}_{E}(B) + \frac{\mu^2}{n^2 \cdot 2^n} \ .$$

Furthermore, B runs in time t and makes at most $\lfloor(\mu + q \cdot (2n-1)/n)\rfloor$ oracle queries, totaling at most $\mu + q \cdot (2n-1)$ bits.

The following theorem (which is a modification of Theorem 7.1 from [15]) states that the encryption scheme of Construction 3.2.2 is IND-CPA, if the underlying encoding scheme is Coll-CPA, the underlying MAC scheme is PRF, and the encryption scheme of Construction 3.2.4 is IND-CPA.

Theorem 3.2.7. Let $\mathcal{SE} = (\mathcal{K}_e, \mathcal{E}, \mathcal{D})$, $\mathcal{MAC} = (\mathcal{K}_t, \mathcal{T})$, and $\mathcal{EC} = (\mathit{Encode}, \mathit{Decode})$ be an encryption scheme, a MAC, and an encoding scheme, respectively, such that the outputs of the encoding scheme are compatible with the inputs to $\mathcal{E}, \mathcal{T}$. Let $\mathcal{SE}'$ and $\mathcal{SE}''$ be the associated encryption schemes as per Construction 3.2.2 and Construction 3.2.4, respectively. For any adversary S attacking IND-CPA security of $\mathcal{SE}'$, that runs in time t and makes q queries to the left-right encryption oracle, totaling μ bits, there exist adversaries A attacking IND-CPA security of $\mathcal{SE}''$, G attacking PRF security of $\mathcal{MAC}$, and C attacking Coll-CPA security of $\mathcal{EC}$, such that

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}'}(S) \leq \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}''}(A) + 2 \cdot \mathbf{Adv}^{\text{prf}}_{\mathcal{MAC}}(G) + 2 \cdot \mathbf{Adv}^{\text{coll-cpa}}_{\mathcal{EC}}(C) \tag{7}$$

Furthermore, A and C use the same resources as S, while G runs in time t and makes q oracle queries, totaling at most $(\mu + q \cdot (2n-1))$ bits.

Proof of Theorem 3.2.7. The proof is very similar to the Proof of Theorem 7.1 of [15]. Let S be an adversary attacking IND-CPA security of $\mathcal{SE}'$. For $x \in \{1, 2, 3\}$, we define the following experiments associated with S:

Experiment $\mathbf{ExpH}_x$

$K_e \xleftarrow{\$} \mathcal{K}_e,\ K_t \xleftarrow{\$} \mathcal{K}_t$

Run S, replying to its oracle query (M_0, M_1) as follows:

$(M_{e,0}, M_{t,0}) \xleftarrow{\$} Encode(M_0),\ (M_{e,1}, M_{t,1}) \xleftarrow{\$} Encode(M_1)$

Switch(x)

Case $x = 1$: $C \leftarrow \mathcal{E}_{K_e}(M_{e,1}),\ \sigma \leftarrow \mathcal{T}_{K_t}(M_{t,1})$

Case $x = 2$: $C \leftarrow \mathcal{E}_{K_e}(M_{e,0}),\ \sigma \leftarrow \mathcal{T}_{K_t}(M_{t,1})$

Case $x = 3$: $C \leftarrow \mathcal{E}_{K_e}(M_{e,0}),\ \sigma \leftarrow \mathcal{T}_{K_t}(M_{t,0})$

Return $C||\sigma$ to S.

Until S halts and returns a bit b.

Return b.

For $x \in \{1, 2, 3\}$, let $P_x = \Pr[\ \mathbf{ExpH}_x = 1\ ]$ denote the probability that experiment $\mathbf{ExpH}_x$ returns 1. By the definition of $\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}'}(S)$, we have

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}'}(S) = P_1 - P_3 = (P_1 - P_2) + (P_2 - P_3) \tag{8}$$

Given S, there exist adversaries A, G and C, such that the following lemmas hold, and these adversaries use the resources specified in Theorem 3.2.7.

Lemma 3.2.8. $P_1 - P_2 \leq \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}''}(A)$.

Lemma 3.2.9. $P_2 - P_3 \leq 2 \cdot \mathbf{Adv}^{\text{prf}}_{\mathcal{MAC}}(G) + 2 \cdot \mathbf{Adv}^{\text{coll-cpa}}_{\mathcal{EC}}(C)$.

Eq. (8), and the above lemmas imply Theorem 3.2.7. □

Proof of Lemma 3.2.8. We construct an adversary A attacking IND-CPA security of $\mathcal{SE}''$, using the adversary S. A first runs $\mathcal{K}_t$ to obtain a key K_t. For every message-pair query (M_0, M_1) that S makes, A uses that message-pair to query to its own oracle and gets back $conf||C$. Now, it pads M_1 to multiple block length and computes $M_{t,1} \leftarrow conf||M_1$, $\sigma \leftarrow \mathcal{T}_{K_t}(M_{t,1})$. It then gives $C||\sigma$ to S. When S halts and returns a bit b', A halts and returns b'.

If $b = 1$, the adversary A simulates S in the exact same environment as that of $\mathbf{ExpH}_1$. Similarly, if $b = 0$, the adversary A simulates S in the exact same environment as that of $\mathbf{ExpH}_2$. Thus,

$$\begin{aligned} P_1 - P_2 &= \Pr\left[\, \mathbf{Exp}^{\text{ind-cpa-1}}_{\mathcal{SE}''}(A) = 1 \,\right] - \Pr\left[\, \mathbf{Exp}^{\text{ind-cpa-0}}_{\mathcal{SE}''}(A) = 1 \,\right] \\ &= \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}''}(A). \end{aligned}$$

Adversary A uses the same resources as S. □

Proof of Lemma 3.2.9. The proof follows directly from Lemma 7.7 and Theorem 7.4 of [15]. □

Below we claim that the encoding scheme $\mathcal{EC}$ in the Simplified profile is Coll-CPA.

Claim 3.2.10. For any adversary C making q queries to the encoding oracle $\mathcal{EC}(\cdot)$,

$$\mathbf{Adv}^{\text{coll-cpa}}_{\mathcal{EC}}(C) \leq \frac{q(q-1)}{2^{n+1}} .$$

Proof of Claim 3.2.10. To justify the claim, we note that *Encode* algorithm prepends a random n-bit confounder to the message, and the only chance that the adversary can make any two encodings M_t, M'_t collide is if any two of q confounders happen to be the same. This can happen with probability at most $\frac{q(q-1)}{2^{n+1}}$, by the well-known birthday bound. □

Theorem 3.2.7, Claim 3.2.6, and Claim 3.2.10 imply the following.

Claim 3.2.11. The authenticated encryption scheme $\mathcal{SE}'$ described by the Simplified profile (Construction 3.2.2) is IND-CPA secure, if the underlying blockcipher E is a PRF, and the underlying MAC is a PRF.

Concretely, for any adversary S attacking IND-CPA security of $\mathcal{SE}'$, that runs in time t, and makes q queries to the left-right encryption oracle, totaling μ bits, there exist adversaries B and G attacking PRF security of E and $\mathcal{MAC}$, respectively, such that

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}'}(S) \leq \mathbf{Adv}^{\text{prf}}_{E}(B) + 2 \cdot \mathbf{Adv}^{\text{prf}}_{\mathcal{MAC}}(G) + \frac{q(q-1)}{2^{n+1}} + \frac{\mu^2}{n^2 \cdot 2^{n+1}} .$$

Furthermore, B runs in time t and makes at most $\lfloor(\mu + q \cdot (2n - 1)/n)\rfloor$ oracle queries, totaling at most $\mu + q \cdot (2n - 1)$ bits; G runs in time t and makes q oracle queries, totaling at most $\mu + q \cdot (2n - 1)$ bits.

IND-CCA SECURITY. Eq. (5), Claim 3.2.11, Theorem 2.2.4, and Theorem 2.3.5 imply Eq. (6).

□

CHAPTER IV

RANDOMNESS GENERATION IN KERBEROS

In this chapter, we present our hash-function-based pseudorandom generator, whose output can be used in place of random strings, e.g., session keys, sequence numbers, confounders, etc. are all suggested to be generated randomly in Kerberos specifications. In Section 4.1, we recall a general design technique used in several prior works that we call PRG from Iterates. This is followed by our construction in Section 4.2, and its proof of security in Section 4.3. Finally, we discuss some improvements in our assumptions and efficiency in Section 4.4 and Section 4.5, respectively.

4.1 PRG from Iterates

Most of the pseudorandom generators (PRGs) that we know today employ a general design technique: take a function that remains one-way on iterates, and iterate that function for a desired number of times, extracting hardcore bits at every iteration. Below we give a general theorem for the security of such PRGs. The theorem already exists in some form in the cryptographic literature (or, is implied from results in several papers, [67, 48, 55], to name a few), but we restate it and sketch its proof here for two main reasons. One is that the proof has evolved over time, starting from Levin's work [67], followed by a proof sketch by Goldreich et al. (cf. Appendix B in [48]), and the improved construction of hard-core predicate by Goldreich and Levin [49]. The second reason is that none of the prior works state the result in its entirety with a concrete security statement.

We will start with a more general definition that also captures the definition of pseudorandomness presented in Section 2.6. Let $\mathcal{X}$ and $\mathcal{Y}$ be random variables with equal output lengths. Let D be an adversary for distinguishing $\mathcal{X}$ from $\mathcal{Y}$. The *indistinguishability*

advantage of D, $\mathbf{Adv}^{\text{ind}}_{\mathcal{X},\mathcal{Y}}(D)$ is defined as

$$\mathbf{Adv}^{\text{ind}}_{\mathcal{X},\mathcal{Y}}(D) = \Pr\left[\, x \xleftarrow{\$} \mathcal{X} :\ D(x) = 1 \,\right] \ -\ \Pr\left[\, y \xleftarrow{\$} \mathcal{Y} :\ D(y) = 1 \,\right] .$$

For any adversary P and any pseudorandom generator $\mathcal{G}$, $\mathbf{Adv}^{\text{ind}}_{\mathcal{G},U_{|\mathcal{G}|}}(P) = \mathbf{Adv}^{\text{prg}}_{\mathcal{G}}(P)$, where $U_{|\mathcal{G}|}$ is a uniform distribution of size equal to the output size of $\mathcal{G}$. The following theorem states the security of a PRG constructed from a function that is one-way on iterates, using the well known Goldreich-Levin hardcore bits.

Theorem 4.1.1. Let $f : \{0,1\}^m \to \{0,1\}^n$ be any function, and for any $i \in \mathbb{N}$, let f^i denote its i^{th} iterate, defined arbitrarily but satisfying the following condition: given only $f^i(x)$ for any $x \in \{0,1\}^m$, $f^{i+1}(x)$ should be efficiently computable. For any $k \in \mathbb{N}$, if f^k is one-way on iterates[1], then for random $x, r \in \{0,1\}^m$, the random variables

$$\mathcal{X} = \langle x, r\rangle \,||\, \left\langle f^1(x), r\right\rangle \,||\, \ldots \,||\, \left\langle f^{k-1}(x),\ r\right\rangle ||r||f^k(x) \quad \text{and} \quad \mathcal{Y} = U_k||r||f^k(x)$$

are indistinguishable, where U_k is a uniform distribution of k bits. More formally, for an adversary D with running time t_D, there exists an adversary I with running time t_I, so that

$$\mathbf{Adv}^{\text{ind}}_{\mathcal{X},\mathcal{Y}}(D) \le 8k \cdot \max_{i=1}^{k} \left(\mathbf{Adv}^{\text{htc}}_{f^i,f}(I)\right), \text{ and } t_I = \mathcal{O}\left(m^3 \cdot t_D \cdot \left(\mathbf{Adv}^{\text{ind}}_{\mathcal{X},\mathcal{Y}}(D)\right)^{-4}\right).$$

Informally, the above theorem states that $\langle x, r\rangle \,||\, \left\langle f^1(x), r\right\rangle \,||\, \ldots \,||\, \left\langle f^{k-1}(x),\ r\right\rangle$ is pseudorandom, given r and $f^k(x)$.

Proof Sketch of Theorem 4.1.1. Any adversary trying to distinguish

$$\mathcal{X} = \langle x, r\rangle \,||\, \left\langle f^1(x), r\right\rangle \,||\, \ldots \,||\, \left\langle f^{k-1}(x),\ r\right\rangle ||r||f^k(x) \quad \text{from} \quad \mathcal{Y} = U_k||r||f^k(x)\ ,$$

can distinguish them only from the first k bits, because the remaining portion of $\mathcal{X}$ and $\mathcal{Y}$ are the same. The proof is presented in three parts. In the first part, we show that the indistinguishability of $\mathcal{X}$ and $\mathcal{Y}$ follows from the unpredictability of $\mathcal{X}' = f^k(x)||r||\left\langle f^{k-1}(x), r\right\rangle \,||\, \ldots \,||$

[1] f^k is one-way on iterates, if given $f^k(x)$ for a random $x \in \{0,1\}^m$, it is hard to compute $x' \in \{0,1\}^m$ such that $f(x') = f^k(x)$.

$\left\langle f^1(x),\ r\right\rangle \| \langle x, r\rangle$ (without loss of generality, the output of $\mathcal{X}$ is written in reverse order). Yao [91] showed using hybrid argument that a sequence is indistinguishable from random, if and only if, it is hard to predict the next bit of the sequence, for every prefix of the sequence. Using this result, for an adversary D with running time t_D, there exist an adversary U with running time t_U, and $i \in [k-1]$, such that given $\mathcal{X}'_i = f^k(x)\|r\| \left\langle f^{k-1}(x), r\right\rangle \| \ldots \| \left\langle f^i(x),\ r\right\rangle$, U can output the next bit $\left\langle f^{i-1}(x),\ r\right\rangle$, so that

$$\left(\Pr\left[\ U\left(\mathcal{X}'_i\right) = \left\langle f^{i-1}(x),\ r\right\rangle\ \right] - \frac{1}{2}\right) \geq \frac{\mathbf{Adv}^{\mathrm{ind}}_{\mathcal{X},\mathcal{Y}}(D)}{2k},\ \text{and } t_U \approx t_D,$$

where $f^0(x) = x$.

In the second part, we show that given the adversary U with running time t_U, there exists an adversary A with running time t_A that can distinguish the hardcore predicate $b\left(f^{i-1}(x), r\right) = \left\langle f^{i-1}(x),\ r\right\rangle$ from random, given $\widehat{f^i}(x, r) = \left(f^i(x), r\right)$, so that

$$\mathbf{Adv}^{\mathrm{hcp}}_{\widehat{f^i},b}(A) = \left(\Pr\left[\ U\left(\mathcal{X}'_i\right) = \left\langle f^{i-1}(x),\ r\right\rangle\ \right] - \frac{1}{2}\right),\ \text{and } t_A \approx t_U.$$

A is easy to construct. We know from the theorem that $f^{i+1}(x), \ldots, f^k(x)$ are efficiently computable from $f^i(x)$, so given $f^i(x)$ and r, A can compute $\mathcal{X}'_i$, which it can use to run U to get back $\left\langle f^{i-1}(x),\ r\right\rangle$.

Finally, using Theorem 2.5.9, given the adversary A with running time t_A, one can construct an adversary I with running time t_I that can compute $f^{i-1}(x)$, given $f^i(x)$, so that

$$\mathbf{Adv}^{\mathrm{htc}}_{f^i,f}(I) \geq \frac{\mathbf{Adv}^{\mathrm{hcp}}_{\widehat{f^i},b}(A)}{4},\ \text{and } t_I = O\left(m^3 \cdot t_A \cdot \left(\mathbf{Adv}^{\mathrm{hcp}}_{\widehat{f^i},b}(A)\right)^{-4}\right).$$

Putting things together, we have

$$\max_{i=1}^{k}\left(\mathbf{Adv}^{\mathrm{htc}}_{f^i,f}(I)\right) \geq \frac{\mathbf{Adv}^{\mathrm{ind}}_{\mathcal{X},\mathcal{Y}}(D)}{8k}\ ,\ \text{ and }\ t_I = O\left(m^3 \cdot t_D \cdot \left(\mathbf{Adv}^{\mathrm{ind}}_{\mathcal{X},\mathcal{Y}}(D)\right)^{-4}\right)\ .$$

□

4.2 Our PRG Construction

We first define the *subset iterate*, a particular way to iterate a hash function on a subset of the actual domain. We use this in our PRG construction.

Subset Iterate. Let $\mathcal{H}$ be a hash function family, where each $h \in \mathcal{H}$ is a mapping from $\{0,1\}^m$ to $\{0,1\}^n$. For any $i \in \mathbb{N}$ and any $h \in \mathcal{H}$, we define the i^{th} *subset iterate* of h, h^i, and denote the corresponding family by $\mathcal{H}^i$. For $x \in \{0,1\}^n$, h^i is defined recursively as

$$\begin{aligned} h^1(x) &= h(x \| 0^{m-n}) \ , \\ h^i(x) &= h\left(h^{i-1}(x) \| 0^{m-n}\right) \quad \forall i > 1 \ . \end{aligned}$$

Any unambiguous padding (in place of zeroes, above) can be used to make the input to h of size m bits. For any $i \in \mathbb{N}$, we define the *one-way on iterates or owi* advantage of an adversary I attacking $\mathcal{H}^i$, $\mathbf{Adv}^{\text{owi}}_{\mathcal{H}^i}(I)$ as

$$\Pr\left[h \xleftarrow{\$} \mathcal{H}, x \xleftarrow{\$} \{0,1\}^n, x' \xleftarrow{\$} I\left(h, h^i(x)\right) : \ h(x' \| 0^{m-n}) = h^i(x) \right].$$

4.2.1 The Scheme

We now present our PRG construction. It requires a very short seed (twice the output size of the hash function).

Construction 4.2.1. Let $\mathcal{H}$ be a hash function family, where each $h \in \mathcal{H}$ is a mapping from $\{0,1\}^m$ to $\{0,1\}^n$. For any $l > 2n$, a random $h \in \mathcal{H}$, which we assume becomes publicly known, and a random seed $s \in \{0,1\}^{2n}$, the pseudorandom generator $\mathcal{G}$ parses the input s as $x \| r$, such that both x and r are n-bit strings, and outputs

$$\langle x, r \rangle \| \left\langle h^1(x), r \right\rangle \| \ldots \| \left\langle h^{l-n-1}(x), r \right\rangle \| r \ ,$$

where for two bitstrings x $(= x_1 \| \ldots \| x_n)$ and r $(= r_1 \| \ldots \| r_n)$, $\langle x, r \rangle = \sum_{i=1}^{n} x_i \cdot r_i \pmod 2$ is their inner product modulo 2.

Note that the seed length of $\mathcal{G}$ is $2n$, and it is independent of the output length l. We now present the security analysis of the above construction.

4.2.2 Security

For simplicity, in the following theorem we assume that the underlying hash function family is regular. We will show how to relax this assumption to worst-case regularity in Section 4.4.

Theorem 4.2.2. Let $\mathcal{H}$ be a hash function family, where each $h \in \mathcal{H}$ is a regular function from $\{0,1\}^m$ to $\{0,1\}^n$ and takes time $t_{\mathcal{H}}$ in computation. For any $l > 2n$, let $\mathcal{G}$ be the associated PRG, as defined by Construction 4.2.1. Then for an adversary P attacking the pseudorandomness of $\mathcal{G}$ with running time t_P, there exists an adversary C attacking the collision-resistance of $\mathcal{H}$ with running time t_C, and $q = \lfloor t_C/t_{\mathcal{H}} \rfloor$, so that

$$\mathbf{Adv}^{\mathrm{prg}}_{\mathcal{G}}(P) \leq 24 \cdot (l-n) \cdot \left[\binom{\lfloor q/(l-n) - 2 \rfloor}{2}^{-1} \cdot 2^n \cdot \left(\mathbf{Adv}^{\mathrm{cr}}_{\mathcal{H}}(C) \right)^2 \right]^{\frac{1}{3}},$$

$$\text{and } t_C = \max \left\{ O\left(n^3 \cdot t_P \cdot \left(\mathbf{Adv}^{\mathrm{prg}}_{\mathcal{G}}(P) \right)^{-4} \right), 2(l-n)t_{\mathcal{H}} \right\}.$$

Remark. The above theorem gives a relation between the pseudorandomness of $\mathcal{G}$ and the collision-resistance of its underlying hash function $\mathcal{H}$. The advantage equation provides meaningful security guarantees only if $\mathbf{Adv}^{\mathrm{cr}}_{\mathcal{H}}(C) < 2^{-n/2}$. Also, as pointed out in the proof of Theorem 4.3.1, the above advantage expression can be made tighter (i.e., $\left(\mathbf{Adv}^{\mathrm{cr}}_{\mathcal{H}}(C) \right)^2$ could be replaced with $\mathbf{Adv}^{\mathrm{tcr}}_{\mathcal{H}}(C_1) \cdot \mathbf{Adv}^{\mathrm{cr}}_{\mathcal{H}}(C_2)$ for C_1, C_2 attacking the target collision-resistance and collision-resistance of $\mathcal{H}$, respectively), though the expression would become even more complicated. We present the proof in Section 4.3.

4.3 Proof of Theorem 4.2.2

We start with a short overview of the proof. The proof consists of two main parts: first we prove that the subset iterate used in the construction of our PRG is one-way on iterates (Theorem 4.3.1), and then we use the general result of Levin [67] (Theorem 4.1.1) to show that our PRG is secure.

The subset iterate is constructed using a hash function. Now, suppose that we have an algorithm I that can invert the subset iterate, i.e. given $(h, h^i(x))$ for any $i \geq 2$, random h, and random x, it returns x' such that $h(x'||0^{m-n}) = h^i(x)$. Then, we can use I to break the target collision-resistance (TCR) of the underlying hash function. The challenge for the TCR attack (h, x) is used to compute $h(x)$, and then $(h, h(x))$ given to I, and assuming that $h(x) \in \mathsf{Im}(h^i)$, with a very high probability the output of I, x' (and x) is a collision instance for h. These steps are similar to those in the proof from [55].

Now, the main challenge is to show that with a non-negligible probability $h(x) \in \mathsf{Im}(h^i)$ (Lemma 4.3.4). The proof of the above is the crux and the main novelty of our analysis. We basically show that on iteration, the image set of the subset iterate shrinks by only a polynomial fraction, i.e., for any $i \geq 2$, $|\mathsf{Im}(h^i)|/|\mathsf{Im}(h^{i-1})|$ is a polynomial fraction. For this we rely on Lemma 4.3.2, which says that the collision probability (in the birthday attack) of a subset iterate degrades only by a multiplicative factor of the square of the number of iterations. It may not be obvious, but the size of the image set and the collision probability of any function are closely related, which is precisely the reason why we are able to prove Lemma 4.3.2.

Before we provide the full security proof, we present some justification for our approach. One could argue that it is better to directly assume that the underlying CRHF is one-way on iterates (OWI), and be done with it. We don't take this approach for the following reasons. First, the OWI property appears to be hard to test in practice. Unlike collision-resistance, we do not know of any experiment carried out by practitioners to measure the strength of a function against this kind of attack. Second, we do not know how does OWI security degrade with the number of iterations, which may be crucial in finding out exactly how many bits can be generated securely by any PRG.

In order to prove Theorem 4.2.2, we state the following theorem about the OWI security of the subset iterate used in the construction of our PRG. This theorem together with

Theorem 4.1.1 (by substituting $(l - n)$ for k) will imply Theorem 4.2.2. (One might notice some inconsistencies between Theorem 4.3.1 and Theorem 4.1.1 in the sense that the underlying primitive in the former is a function family, while it is only a function in the latter. We note, however, that Theorem 4.1.1 is applicable without any change in the security reduction to our PRG construction from a hash function family.)

Theorem 4.3.1. Let $\mathcal{H}$ be a hash function family, where each $h \in \mathcal{H}$ is a regular function from $\{0,1\}^m$ to $\{0,1\}^n$ and takes time $t_{\mathcal{H}}$ in computation. For any $i \in \mathbb{N}$, let $\mathcal{H}^i$ be the associated i^{th} subset iterate function family of $\mathcal{H}$, as defined in Section 4.2. Then for an adversary I with running time t_I, there exists an adversary C with running time t_C, and $q = \lfloor t_C / t_{\mathcal{H}} \rfloor$, so that

$$\mathbf{Adv}^{\mathrm{owi}}_{\mathcal{H}^i}(I) \leq 3 \cdot \left[\binom{\lfloor q/i - 2 \rfloor}{2}^{-1} \cdot 2^n \cdot \left(\mathbf{Adv}^{\mathrm{cr}}_{\mathcal{H}}(C) \right)^2 \right]^{\frac{1}{3}}, \text{ and } t_C = \max\{t_I, 2it_{\mathcal{H}}\}.$$

Proof of Theorem 4.3.1. We construct an adversary C_1 with running time $t_{C_1} = t_I$, for attacking the target collision-resistance of $\mathcal{H}$. C_1 is given a random $h \in \mathcal{H}$ and a random $x \in \{0,1\}^m$. It runs the adversary I attacking one-wayness on iterates of $\mathcal{H}^i$ with input $(h, h(x))$. Let x' be the output of I. If $x \neq x'||0^{m-n}$ and $h(x) = h(x'||0^{m-n})$, it returns $x'||0^{m-n}$.

We state the following three lemmas from which we will derive the inequality of Theorem 4.3.1. Lemma 4.3.2 gives an upper bound on the collison probability of birthday attack on the subset iterate of a hash function family. Lemma 4.3.3 which is similar to Claim 3.3 of [55], states that the set of inputs on which the adversary I succeeds reasonably well (better than one third of its advantage) is not small (at least two thirds of its advantage) in size. And, Lemma 4.3.4 which is similar to Lemma 3.4 of [55], states that the set of inputs that I should get in the actual experiment ($h^i(x)$ for a random $x \in \{0,1\}^n$) and the set of inputs that it actually gets in the above experiment simulated by C_1 ($h(x)$ for a random $x \in \{0,1\}^m$), overlap for the most part.

Lemma 4.3.2. Let $\mathcal{H}$ be a hash function family, where each $h \in \mathcal{H}$ is a mapping from $\{0,1\}^m$ to $\{0,1\}^n$ and takes time $t_{\mathcal{H}}$ in computation. For any $i \in \mathbb{N}$, let $\mathcal{H}^i$ be the associated

i^{th} subset iterate of $\mathcal{H}$, as defined in Section 4.2. Then for any $q \geq 2i$, there exists an adversary C_2 that runs in time (at most) $q \cdot t_{\mathcal{H}}$, such that

$$\mathbf{CP}(\mathcal{H}^i, 2) \leq \frac{\mathbf{Adv}^{\mathrm{cr}}_{\mathcal{H}}(C_2)}{\binom{\lfloor q/i-2 \rfloor}{2}} .$$

Proof of Lemma 4.3.2. We know that for any function f with output size n bits and balance measure $\mu(f)$, (upto constant factors) the collision probability for any $t \in \mathbb{N}$ trials, $\mathbf{CP}(f, t) = \binom{t}{2} \cdot 2^{-n\mu(f)}$, see [14] for details. Let $q' = \lfloor q/i - 2 \rfloor$, then

$$\mathbf{CP}(\mathcal{H}^i, 2) = \frac{\mathbf{CP}(\mathcal{H}^i, q')}{\binom{q'}{2}} .$$

Also, it is immediate that there exists an adversary C' running in time equivalent to q' computations of $h^i \in \mathcal{H}^i$, such that

$$\mathbf{Adv}^{\mathrm{cr}}_{\mathcal{H}^i}(C') \geq \mathbf{CP}(\mathcal{H}^i, q') .$$

(In the worst case, C' could simply run the birthday attack with q' trials.)

Now, given C' we will construct the adversary C_2 (from the lemma) that runs in time at most $q \cdot t_{\mathcal{H}}$, so that

$$\mathbf{Adv}^{\mathrm{cr}}_{\mathcal{H}}(C_2) = \mathbf{Adv}^{\mathrm{cr}}_{\mathcal{H}^i}(C') .$$

Note that for any $h^i \in \mathcal{H}^i$, and any $x \neq x' \in \{0,1\}^n$, if $h^i(x) = h^i(x')$, then there exists $j < i$, such that $h^j(x) \neq h^j(x')$ and $h^{j+1}(x) = h^{j+1}(x')$. When C' returns (x, x'), C_2 computes $y \leftarrow h^j(x)$, $y' \leftarrow h^j(x')$, and returns $(y\|0^{m-n}, y'\|0^{m-n})$. Recall that $y \neq y'$ and $h(y\|0^{m-n}) = h(y'\|0^{m-n})$, so the advantage of C_2 is the same as that of C'. Assuming that one computation of $h^i \in \mathcal{H}^i$ requires the same time as i computations of $h \in \mathcal{H}$, we have that the running time of C_2 is at most $q \cdot t_{\mathcal{H}}$ ($\geq (i \cdot q' + 2i) \cdot t_{\mathcal{H}}$), because apart from running C' (which is equivalent to $i \cdot q'$ computations of $h \in \mathcal{H}$), C_2 does $2j$ ($< 2i$) computations of $h \in \mathcal{H}$ to compute its own output. Thus, $\mathbf{Adv}^{\mathrm{cr}}_{\mathcal{H}}(C_2)$ is equal to

$$\mathbf{Adv}^{\mathrm{cr}}_{\mathcal{H}^i}(C') \geq \mathbf{CP}(\mathcal{H}^i, q') = \mathbf{CP}(\mathcal{H}^i, 2) \cdot \binom{q'}{2} = \mathbf{CP}(\mathcal{H}^i, 2) \cdot \binom{\lfloor q/i - 2 \rfloor}{2},$$

from which the lemma follows. □

Lemma 4.3.3. Let $\mathcal{H}$ be a hash function family, where each $h \in \mathcal{H}$ is a mapping from $\{0,1\}^m$ to $\{0,1\}^n$. For any $i \in \mathbb{N}$ and any $h \in \mathcal{H}$, let h^i be the associated i^{th} subset iterate and $\mathcal{H}^i$ be the corresponding family, as defined in Section 4.2. For any adversary I, consider the following probabilities in an experiment where a random $h \in \mathcal{H}$ and a random $x \in \{0,1\}^n$ are picked, and a set $S \subseteq \mathsf{Im}(h^i)$ is defined as

$$S = \left\{ y \in \mathsf{Im}(h^i) : \ \Pr\left[\, h\left(I\left(h, y\right)\right) = y \,\right] > \frac{1}{3} \cdot \mathbf{Adv}^{\mathrm{owi}}_{\mathcal{H}^i}(I) \right\}.$$

Then,

$$\Pr\left[\, h^i(x) \in S \,\right] \geq \frac{2}{3} \cdot \mathbf{Adv}^{\mathrm{owi}}_{\mathcal{H}^i}(I).$$

Proof of Lemma 4.3.3. Assume (for contradiction) that in the above experiment, where a random $h \in \mathcal{H}$ and a random $x \in \{0,1\}^n$ are picked, and a set $S \subseteq \mathsf{Im}(h^i)$ is defined as above, the following holds:

$$\Pr\left[\, h^i(x) \in S \,\right] < \frac{2}{3} \cdot \mathbf{Adv}^{\mathrm{owi}}_{\mathcal{H}^i}(I).$$

Then we have

$$\begin{aligned} \mathbf{Adv}^{\mathrm{owi}}_{\mathcal{H}^i}(I) &\leq \Pr\left[\, h^i(x) \in S \,\right] \cdot 1 + \Pr\left[\, h^i(x) \notin S \,\right] \cdot \frac{1}{3} \cdot \mathbf{Adv}^{\mathrm{owi}}_{\mathcal{H}^i}(I) \\ &< \frac{2}{3} \cdot \mathbf{Adv}^{\mathrm{owi}}_{\mathcal{H}^i}(I) + \frac{1}{3} \cdot \mathbf{Adv}^{\mathrm{owi}}_{\mathcal{H}^i}(I), \end{aligned}$$

where the probabilities are over randomly picked $h \in \mathcal{H}$ and $x \in \{0,1\}^n$.

S is the set of points where the adversary's advantage is greater than one-third of its actual (or, average) advantage. So, setting the adversary's advantage to be 1 for points inside S and one-third for points outside S, we get the first inequality. The second inequality follows directly from the above assumption.

Thus, $\mathbf{Adv}^{\mathrm{owi}}_{\mathcal{H}^i}(I) < \mathbf{Adv}^{\mathrm{owi}}_{\mathcal{H}^i}(I)$, which is a contradiction. □

Lemma 4.3.4. Let $\mathcal{H}$ be a hash function family, where each $h \in \mathcal{H}$ is a mapping from $\{0,1\}^m$ to $\{0,1\}^n$. For any $i \in \mathbb{N}$ and any $h \in \mathcal{H}$, let h^i be the associated i^{th} subset iterate and $\mathcal{H}^i$ be the corresponding family, as defined in Section 4.2. Consider the following

probabilities in an experiment where a random $h \in \mathcal{H}$ and a random $x \in \{0,1\}^n$ are picked. If for any $T \subseteq \mathsf{Im}(h^i)$ and any $\delta \in [0,1]$,

$$\Pr\Big[\ h^i(x) \in T\ \Big] \geq \delta,$$

then

$$\Pr[\ h(x) \in T\ ] \geq \frac{\delta^2}{2^{n+1} \cdot \mathbf{CP}(h^i, 2)}.$$

Proof of Lemma 4.3.4. We will first compute a lower bound on the collision probability of h^i for two trials, $\mathbf{CP}(h^i, 2)$. Pick two elements x_1, x_2 uniformly at random from $\{0,1\}^n$, and then compute the probability that both $h^i(x_1), h^i(x_2)$ are equal and belong to the set T. This probability is clearly a lower bound on $\mathbf{CP}(h^i, 2)$, because T is a subset of $\mathsf{Im}(h^i)$. The probability that both $h^i(x_1), h^i(x_2) \in T$ is at least δ^2, and given that $h^i(x_1), h^i(x_2) \in T$, the probability that $h^i(x_1) = h^i(x_2)$ is at least $1/|T|$. The reason is that even though x_1, x_2 are uniformly random elements in $\{0,1\}^n$, $h^i(x_1), h^i(x_2)$ may not[2] be uniformly random elements in T. So, the probability that $h^i(x_1) = h^i(x_2)$ can be lower bounded by computing the probability of getting the same element, when two elements are picked (with replacement) uniformly at random from the set T. By simple probability theory, the probability of such an event is $1/|T|$. It may however be noted that in the above calculation, we are also counting trivial collisions, i.e. when $x_1 = x_2$. To compensate for this, we subtract 2^{-n} from the above probability. Hence,

$$\mathbf{CP}(h^i, 2) \geq \frac{\delta^2}{|T|} - \frac{1}{2^n}. \tag{9}$$

From Eq. (9), we have

$$|T| \geq \frac{\delta^2}{\mathbf{CP}(h^i, 2) + 2^{-n}} \geq \frac{\delta^2}{2 \cdot \mathbf{CP}(h^i, 2)},$$

because $\mathbf{CP}(h^i, 2) \geq 2^{-n}$.

[2]These elements are uniformly distributed, only if h^i is a regular function.

For any $h \in \mathcal{H}$, $\mathsf{Im}(h^i) \subseteq \mathsf{Im}(h)$, and since $T \subseteq \mathsf{Im}(h^i)$, we have that $T \subseteq \mathsf{Im}(h)$. Also, since h is a regular function[3] and $\mathsf{Im}(h) \leq 2^n$, we have that

$$\Pr\left[h \xleftarrow{\$} \mathcal{H},\ x \xleftarrow{\$} \{0,1\}^m :\ h(x) \in T \right] = \frac{|T|}{|\mathsf{Im}(h)|} \geq \frac{|T|}{2^n}. \tag{10}$$

Thus, the statement of the lemma follows. □

Implication of Lemma 4.3.2, Lemma 4.3.3, and Lemma 4.3.4. Substituting S for T and $\frac{2}{3} \cdot \mathbf{Adv}^{\mathrm{owi}}_{\mathcal{H}^i}(I)$ (from Lemma 4.3.3) for δ in Lemma 4.3.4, we get that for a random $h \in \mathcal{H}$, adversary I and subset S as defined in Lemma 4.3.3

$$\begin{aligned}\Pr\left[h \xleftarrow{\$} \mathcal{H},\ x \xleftarrow{\$} \{0,1\}^m :\ h(x) \in S \right] &\geq \frac{\left(\frac{2}{3} \cdot \mathbf{Adv}^{\mathrm{owi}}_{\mathcal{H}^i}(I)\right)^2}{2^{n+1} \cdot \mathbf{CP}(h^i, 2)} \\ &\geq \frac{2^2}{3^2} \cdot \frac{\left(\mathbf{Adv}^{\mathrm{owi}}_{\mathcal{H}^i}(I)\right)^2}{2^{n+1} \cdot \mathbf{CP}(h^i, 2)}.\end{aligned}$$

The above equation is a lower bound on the probability that for a random $h \in \mathcal{H}$ and a random $x \in \{0,1\}^m$, I's challenge, $h(x)$ belongs to the subset S. From the description of C_1, it is clear that $\mathbf{Adv}^{\mathrm{tcr}}_{\mathcal{H}}(C_1)$

$$\begin{aligned}&= \Pr\left[h \xleftarrow{\$} \mathcal{H}, x \xleftarrow{\$} \{0,1\}^m, x' \xleftarrow{\$} I(h, h(x)) : x \neq x' \| 0^{m-n} \bigwedge h(x' \| 0^{m-n}) = h(x) \right] \\ &= \Pr\left[h \xleftarrow{\$} \mathcal{H}, x \xleftarrow{\$} \{0,1\}^m, x' \xleftarrow{\$} I(h, h(x)) : x \neq x' \| 0^{m-n} \mid h(x' \| 0^{m-n}) = h(x) \right] \\ &\quad \times \Pr\left[h \xleftarrow{\$} \mathcal{H}, x \xleftarrow{\$} \{0,1\}^m, x' \xleftarrow{\$} I(h, h(x)) : h(x' \| 0^{m-n}) = h(x) \right].\end{aligned}$$

Let us denote the two probabilities in the last equation by P_1 and P_2, respectively. So, $\mathbf{Adv}^{\mathrm{tcr}}_{\mathcal{H}}(C_1) = P_1 \cdot P_2$. We know that

$$\begin{aligned}P_1 &\geq \Pr\left[z \xleftarrow{\$} \{0,1\}^{m-n} :\ z \neq 0^{m-n} \right] \\ &\geq \frac{2^{m-n} - 1}{2^{m-n}} \geq \frac{1}{2},\end{aligned}$$

because x is a uniformly random m-bit string, so the probability that the last $m - n$ bits of x are all 0's is at most 2^{n-m}. Also, from Lemma 4.3.3, we have that for a random $h \in \mathcal{H}$,

[3] We note that this is the only point in the proof that relies on the assumption that h is a regular function.

adversary I and subset S as defined in Lemma 4.3.3

$$\begin{aligned} P_2 &\geq \Pr\left[h \xleftarrow{\$} \mathcal{H},\ x \xleftarrow{\$} \{0,1\}^m :\ h(x) \in S \right] \cdot \frac{1}{3} \cdot \mathbf{Adv}^{\text{owi}}_{\mathcal{H}^i}(I) \\ &\geq \frac{2^2}{3^2} \cdot \frac{\left(\mathbf{Adv}^{\text{owi}}_{\mathcal{H}^i}(I)\right)^2}{2^{n+1} \cdot \mathbf{CP}(h^i, 2)} \cdot \frac{1}{3} \cdot \mathbf{Adv}^{\text{owi}}_{\mathcal{H}^i}(I) \\ &\geq \frac{2^2}{3^3} \cdot \frac{\left(\mathbf{Adv}^{\text{owi}}_{\mathcal{H}^i}(I)\right)^3}{2^{n+1} \cdot \mathbf{CP}(h^i, 2)} . \end{aligned}$$

The second inequality is from the lower bound on the probability that I's challenge $h(x)$ belongs to the subset S, as computed above. Thus,

$$\mathbf{Adv}^{\text{tcr}}_{\mathcal{H}}(C_1) \geq \frac{\left(\mathbf{Adv}^{\text{owi}}_{\mathcal{H}^i}(I)\right)^3}{3^3 \cdot 2^n \cdot \mathbf{CP}(h^i, 2)} .$$

Combining the above inequality with Lemma 4.3.2, we have that for any $q \geq 2i$, there exists an adversary C_2 that runs in time (at most) $q \cdot t_{\mathcal{H}}$, such that

$$\mathbf{Adv}^{\text{tcr}}_{\mathcal{H}}(C_1) \cdot \mathbf{Adv}^{\text{cr}}_{\mathcal{H}}(C_2) \geq \frac{\binom{\lfloor q/i-2 \rfloor}{2}}{3^3 \cdot 2^n} \cdot \left(\mathbf{Adv}^{\text{owi}}_{\mathcal{H}^i}(I)\right)^3 .$$

Recall that the running time of C_1, $t_{C_1} = t_I$. Let $q = \max\{\lfloor t_I / t_{\mathcal{H}} \rfloor, 2i\}$, and let C denote the adversary (among C_1, C_2) with higher collision-resistance advantage, i.e., $C = C_1$ if $\mathbf{Adv}^{\text{cr}}_{\mathcal{H}}(C_1) \geq \mathbf{Adv}^{\text{cr}}_{\mathcal{H}}(C_2)$, otherwise $C = C_2$. (Note that we are getting rid of *target* collision-resistance advantage for a simpler theorem statement, albeit at a loss in the security guarantee) Then,

$$\left(\mathbf{Adv}^{\text{cr}}_{\mathcal{H}}(C)\right)^2 \geq \frac{\binom{\lfloor q/i-2 \rfloor}{2}}{3^3 \cdot 2^n} \cdot \left(\mathbf{Adv}^{\text{owi}}_{\mathcal{H}^i}(I)\right)^3 .$$

The running time of C, $t_C = \max\{t_I, 2it_{\mathcal{H}}\}$, and hence, Theorem 4.3.1 follows. □

4.4 Relaxing the Regularity Assumption

We introduce a new notion that we call worst-case regularity. It captures the lower bound on the size of the smallest set of preimages of elements from the range of a function. The notion appears somewhat similar to the notions of "weakly regular" introduced by Goldreich et al. [48] and "balance measure" introduced by Bellare and Kohno [14]. However, the

reason for introducing a new notion (instead of working with the previous ones), is that it seems unlikely that one can find a tight relation between worst-case regularity and balance measure (or, weak regularity), and thus a tight bound for our theorem, for any general function (or, a CRHF in particular). The intuition behind this is that while worst-case regularity measures the lower bound on the size of preimages, the other two notions are related to the average of these sizes. We will first present the formal definition of worst-case regularity, and then adjust the statement of our main theorem for the case when the underlying CRHF is not necessarily regular.

Worst-case Regularity. Let $\mathcal{F}$ be a family of functions, where each $f \in \mathcal{F}$ is a mapping from $\{0,1\}^m$ to $\{0,1\}^n$, and let $\alpha \in (0,1]$. We say that $\mathcal{F}$ is α-worst-case regular, if for all $f \in \mathcal{F}$ and all $y \in \mathsf{Im}(f)$

$$|\mathsf{Preim}(f,y)| \geq \alpha \cdot 2^{m-n} \ .$$

For a completely regular function family, $\alpha = 1$.

As pointed out before, the only place where the regularity assumption is required for our proof is in Eq. (10) of Lemma 4.3.4. So, we will first modify this equation and give justification for this modification, and then adjust our main theorem accordingly. For a not-necessarily regular function family Eq. (10) changes as follows.

For any $h \in \mathcal{H}$ and any $T \subseteq \mathsf{Im}(h)$, if $\mathcal{H}$ is α-worst-case regular, then

$$\Pr\left[h \xleftarrow{\$} \mathcal{H},\ x \xleftarrow{\$} \{0,1\}^m :\ h(x) \in T \right] \geq \frac{\alpha \cdot |T|}{2^n} \ , \tag{11}$$

where $\mathcal{H}$ is a hash function family as defined in Lemma 4.3.4. Since $\mathcal{H}$ is α-worst-case regular, the lower bound on the total size of the preimages of elements in T is $(\alpha \cdot 2^{m-n} \cdot |T|)$. So, when an element is picked uniformly at random from a set of size 2^m, the probability that it hits a subset of size $(\alpha \cdot 2^{m-n} \cdot |T|)$ is $\frac{\alpha \cdot |T|}{2^n}$.

Taking the above equation into account, we present the modified main theorem.

Theorem 4.4.1. [**Modified Theorem 4.2.2**] Let $\mathcal{H}$ be an α-worst-case regular hash function family, where each $h \in \mathcal{H}$ is a function from $\{0,1\}^m$ to $\{0,1\}^n$ and takes time $t_{\mathcal{H}}$ in

computation. For any $l > 2n$, let $\mathcal{G}$ be the associated pseudorandom generator, as defined by Construction 4.2.1. Then for an adversary P with running time t_P, there exists an adversary C with running time t_C, and $q = \lfloor t_C/t_{\mathcal{H}} \rfloor$, so that

$$\mathbf{Adv}^{\text{prg}}_{\mathcal{G}}(P) \leq 24 \cdot (l-n) \cdot \left[\binom{\lfloor q/(l-n) - 2 \rfloor}{2}^{-1} \cdot \alpha^{-1} \cdot 2^n \cdot \left(\mathbf{Adv}^{\text{cr}}_{\mathcal{H}}(C) \right)^2 \right]^{\frac{1}{3}},$$

$$\text{and } t_C = \max \left\{ O\left(n^3 \cdot t_P \cdot \left(\mathbf{Adv}^{\text{prg}}_{\mathcal{G}}(P) \right)^{-4} \right), 2(l-n)t_{\mathcal{H}} \right\}.$$

4.5 *Efficiency Improvement*

Instead of extracting just one hardcore bit in an iteration, we can extract upto a constant factor of n hardcore bits, depending on the one-way on iterates security (and hence, the collision-resistance, see Theorem 4.3.1) of the underlying hash function. For the i^{th} iteration, let $\epsilon_i = \max_I \left(\mathbf{Adv}^{\text{owi}}_{\mathcal{H}^i}(I) \right)$ denote the one-way on iterates security of $\mathcal{H}^i$, where the maximum is over all polynomial-time adversary I. Then, one can extract $k_i = O(\log \epsilon_i)$ hardcore bits in the i^{th} iteration without compromising the security of the PRG (cf. Theorem 2.5.6 in [46]). The way to do it is to pick a random r (used with the iterated function's output in the inner product computation) of size $(n + k_i - 1)$ bits, and return $\langle \cdot, r_1 \rangle || \ldots || \langle \cdot, r_k \rangle$, where "$\cdot$" is the output of the function in a particular iteration, and for $j \in [k]$, r_j is the first n bits of r starting from the j^{th} bit. Recall that the same r can be used in all the iterations, so a sufficiently large r ($< 2n$ bits) can be picked in the beginning and used throughout.

CHAPTER V

REVOCATION IN NEWER TYPES OF ENCRYPTION

In this chapter, we present our third and final set of results concerning efficient revocation in IBEs and ABEs. Due to the attractive features provided by these new cryptographic primitives, they have been used in many practical protocols. We present our definitions in Section 5.1, and constructions and their proofs of security in Section 5.2. In Section 5.3, we discuss ways to make our construction secure against chosen-ciphertext attacks. Finally, in Section 5.4, we briefly discuss how to extend our techniques from IBE to ABE.

5.1 Revocable IBE and its Security

We start with defining the general syntax of a Revocable IBE scheme.

5.1.1 Syntax of Revocable IBE

Definition 5.1.1. [Revocable IBE] An *identity-based encryption with efficient revocation* or simply *Revocable IBE* scheme $\mathcal{RIBE} = (\mathcal{S}, \mathcal{SK}, \mathcal{KU}, \mathcal{DK}, \mathcal{E}, \mathcal{D}, \mathcal{R})$ is defined by seven algorithms and has an associated message space MsgSp, an identity space IdSp and a time space TimeSp. We assume that the size of TimeSp is polynomial in the security parameter κ. Each algorithm is run by either one of the three types of parties - key authority, sender or receiver. Key authority maintains a revocation list *rl* and a state *st*. The revocation list *rl* can be part of the state *st*, but we keep it explicit for clarity. In what follows, we call an algorithm stateful only if it updates *rl* or *st*. We treat time as discrete as opposed to continuous.

- The stateful *setup* algorithm $\mathcal{S}$ (run by key authority) takes input the security parameter 1^κ and the number of users n, and outputs a public parameters *pk*, a master key *mk*, a revocation list *rl* (initially empty) and a state *st*.

- The stateful *private key generation* algorithm $\mathcal{SK}$ (run by key authority) takes input the public parameters pk, the master key mk, an identity $\omega \in \mathsf{IdSp}$ and the state st, and outputs a private key sk_ω and an updated state st.

- The *key update generation algorithm* $\mathcal{KU}$ (run by key authority) takes input the public parameters pk, the master key mk, a key update time $t \in \mathsf{TimeSp}$, the revocation list rl and the state st, and outputs a key update ku_t.

- The deterministic *decryption key generation* algorithm $\mathcal{DK}$ (run by receiver) takes input a private key sk_ω and a key update ku_t, and outputs a decryption key $dk_{\omega,t}$, or a special symbol $\perp$ indicating that ω was revoked.

  (We say an identity ω was revoked at time t, if the revocation algorithm $\mathcal{R}$ was run by key authority on input (ω, t, rl, st) for any rl, st.)

- The *encryption* algorithm $\mathcal{E}$ (run by sender) takes input the public parameters pk, an identity $\omega \in \mathsf{IdSp}$, an encryption time $t \in \mathsf{TimeSp}$ and a message $m \in \mathsf{MsgSp}$, and outputs a ciphertext c. For simplicity and w.l.o.g., we assume that the encryption time and identity are efficiently computable from c.

- The deterministic *decryption* algorithm $\mathcal{D}$ (run by receiver) takes input a decryption key $dk_{\omega,t}$ and a ciphertext c, and outputs a message $m \in \mathsf{MsgSp}$, or a special symbol $\perp$ indicating that the ciphertext is invalid.

- The stateful *revocation* algorithm $\mathcal{R}$ (run by key authority) takes input an identity to be revoked $\omega \in \mathsf{IdSp}$, a revocation time $t \in \mathsf{TimeSp}$, the revocation list rl and the state st, and outputs an updated revocation list rl.

The consistency condition requires that for all $\kappa \in \mathbb{N}$ and polynomials (in κ) n, all pk and mk output by setup algorithm $\mathcal{S}$, all $m \in \mathsf{MsgSp}, \omega \in \mathsf{IdSp}, t \in \mathsf{TimeSp}$ and all possible

valid[1] states st and revocation lists rl; if identity ω was not revoked before, or at time t then the following experiment returns 1 with probability 1:

$$(sk_\omega, st) \xleftarrow{\$} \mathcal{SK}(pk, mk, \omega, st)\ ;\ ku_t \xleftarrow{\$} \mathcal{KU}(pk, mk, t, rl, st)$$

$$dk_{\omega,t} \leftarrow \mathcal{DK}(sk_\omega, ku_t)\ ;\ c \xleftarrow{\$} \mathcal{E}(pk, \omega, t, m)$$

If $\mathcal{D}(dk_{\omega,t}, c) = m$ then return 1 else return 0.

Remarks. Note that we differentiate between the terms "private key" and "decryption key". One can also define the decryption key generation algorithm that instead of private key sk_ω, takes input the decryption key for the previous time period $dk_{\omega,t-1}$. We do not discuss this version as it is not used in our construction.

5.1.2 Security of Revocable IBE

We define the *selective-revocable-ID* security for Revocable IBE schemes. Our security model captures the standard notion of selective-ID security, but it also takes into account possible revocations. Since we explicitly consider time periods, in the beginning of the experiment, in addition to the challenge identity the adversary also declares the challenge time. Just as in the standard slective-ID security definition, the adversary can request to learn users' keys. In addition, we let the adversary to revoke users of its choice (including the challenge identity) at any period of time, and see all key updates, which are the decryption key components corresponding to time and are published by key authority for every time period. Unlike in the standard security model, we allow the adversary to learn the private key for the challenge identity, but only if it was revoked prior to, or, at the challenge time. The adversary is given a ciphertext of one of the two messages of its choice, encrypted for the challenge identity and time. It has to guess which one of the two messages was encrypted.

[1] A valid state is the one that is output by either setup algorithm $\mathcal{S}$ or private key generation algorithm $\mathcal{SK}$. A valid revocation list is the one that is output by either $\mathcal{S}$ or $\mathcal{R}$.

First we define (selective) security against chosen-plaintext attack, and then show how to extend the definition to chosen-ciphertext attack.

Definition 5.1.2. [sRID Security] Let $\mathcal{RIBE} = (\mathcal{S}, \mathcal{SK}, \mathcal{KU}, \mathcal{DK}, \mathcal{E}, \mathcal{D}, \mathcal{R})$ be a Revocable IBE scheme. The adversary first outputs the challenge identity and time, and also some information *state* it wants to preserve. Later it is given access to three oracles that correspond to the algorithms of the scheme. The oracles share state.[2] Since we use the simplified notation for the oracles, we define them now:

- The *private key generation* oracle $\mathcal{SK}(\cdot)$ takes input an identity ω, and runs $\mathcal{SK}(pk, mk, \omega, st)$ to return private key sk_ω.
- The *key update generation* oracle $\mathcal{KU}(\cdot)$ takes input a time t, and runs $\mathcal{KU}(pk, mk, t, rl, st)$ to return key update ku_t.
- The *revocation* oracle $\mathcal{R}(\cdot, \cdot)$ takes input an identity ω and a time t, and runs $\mathcal{R}(\omega, t, rl, st)$ to update rl.

For an adversary A and the number of users n, define the following experiments:

Experiment $\mathbf{Exp}^{\text{srid-cpa}}_{\mathcal{RIBE},n}(A)$

$b \xleftarrow{\$} \{0, 1\}$

$(\omega^*, t^*, state) \xleftarrow{\$} A(1^\kappa)$

$(pk, mk, rl, st) \xleftarrow{\$} \mathcal{S}(1^\kappa, n)$

$(m_0, m_1, state) \xleftarrow{\$} A^{\mathcal{SK}(\cdot),\mathcal{KU}(\cdot),\mathcal{R}(\cdot,\cdot)}(pk, state)$

$c^* \xleftarrow{\$} \mathcal{E}(pk, \omega^*, t^*, m_b)$

$d \xleftarrow{\$} A^{\mathcal{SK}(\cdot),\mathcal{KU}(\cdot),\mathcal{R}(\cdot,\cdot)}(pk, c^*, state)$

If $b = d$ return 1 else return 0.

The following conditions must always hold:

[2] To be more formal, we could define a single oracle that maintains the state and invokes these oracles as subroutines. We do not do it for simplicity.

1. $m_0, m_1 \in \mathsf{MsgSp}$ and $|m_0| = |m_1|$.

2. $\mathcal{KU}(\cdot)$ and $\mathcal{R}(\cdot,\cdot)$ can be queried on a time that is greater than or equal to the time of all previous queries, i.e., the adversary is allowed to query only in non-decreasing order of time[3]. Also, $\mathcal{R}(\cdot,\cdot)$ cannot be queried on time t, if $\mathcal{KU}(\cdot)$ was queried on t.[4]

3. If $\mathcal{SK}(\cdot)$ was queried on an identity ω^*, then $\mathcal{R}(\cdot,\cdot)$ must be queried on (ω^*, t) for some $t \leq t^*$.

We define the *advantage* of A, $\mathbf{Adv}^{\text{srid-cpa}}_{\mathcal{RIBE},n}(A)$ as

$$\mathbf{Adv}^{\text{srid-cpa}}_{\mathcal{RIBE},n}(A) = 2 \cdot \Pr\left[\, \mathbf{Exp}^{\text{srid-cpa}}_{\mathcal{RIBE},n}(A) = 1 \,\right] - 1 \ .$$

The scheme $\mathcal{RIBE}$ is said to be *sRID-CPA secure*, if $\mathbf{Adv}^{\text{srid-cpa}}_{\mathcal{RIBE},n}(A)$ is negligible in κ for any efficient A and polynomial n.

Chosen-ciphertext attack. We extend the above definition in a standard way to take into account chosen-ciphertext attacks. Whenever the adversary is given the oracles, it is also given a *decryption oracle* $\mathcal{D}(\cdot)$ that takes input a ciphertext c, and runs $\mathcal{D}(dk_{\omega^*,t}, c)$ to return message m or $\perp$. The usual restriction is that $\mathcal{D}(\cdot)$ cannot be queried on challenge ciphertext c^*. The advantage of the adversary $\mathbf{Adv}^{\text{srid-cca}}_{\mathcal{RIBE},n}(A)$ and *sRID-CCA security* are defined analogously to the CPA setting.

5.2 Our Main Construction

Intuition. At a high level, we build on the (large universe) construction of Fuzzy IBE [83] and the binary tree data structure. We briefly recall the Fuzzy IBE primitive ideas and the basics of the construction.

In the Fuzzy IBE construction from [83], users' keys and ciphertexts are associated with sets of descriptive attributes. A user's key can decrypt a particular ciphertext only if

[3]This is w.l.o.g., because the adversary can query the oracles for all possible time periods one by one.
[4]This is because we assume that the key update is done at the end of the time period t.

some number of attributes (so called "error-tolerance") match between the ciphertext and the key. The number of attributes used to encrypt, and the error-tolerance are fixed during the setup. Security of Fuzzy IBE requires that different users should not be able to pool their attributes together and be able to decrypt a ciphertext which none of them were able to decrypt individually. To prevent collusions, the key generation algorithm of Fuzzy IBE generates a random polynomial (of degree one less than the error-tolerance) for each user. This polynomial is used to compute keys corresponding to a set of attributes. Since all the keys are computed on different polynomials, they cannot be combined in any meaningful way.

In our IBE scheme, messages are encrypted under two "attributes": identity of the receiver and the time period. The decryption key is also computed for attributes identity and time, on a first-degree polynomial, meaning both attributes of the decryption key must match with those of a ciphertext for decryption to be successful. We split the decryption key in two components corresponding to identity and time, that we call private key and key update, respectively. The private key is issued to each user by key authority[5] just like regular private keys in IBE. The key update is published by key authority and is publicly available to all users. To be able to decrypt a ciphertext, a user needs both the private key and the key update. Thus, when key authority needs to revoke a user it may simply stop publishing key updates for that user. As we recalled above, in Fuzzy IBE the polynomial of a decryption key is selected at random to prevent collusion between different keys. Using Fuzzy IBE in a naive way would thus require computing key updates for each user separately. We use a different approach to reduce the number of key updates that key authority needs to compute. We use a binary tree of height h (with at least as many leaves as the number of users in the system) and assign a random polynomial to each node of the tree. Next we associate each user to a unique leaf node. Every user gets keys (corresponding to

[5]We use a different name than PKG to emphasize a new way to handle revocations.

its identity) computed on polynomials of all nodes on the path from the leaf node corresponding to that user to the root node. To be able to decrypt a ciphertext encrypted with time t, a user just needs one key update (corresponding to t) computed on any one of the polynomials of nodes on the path from the leaf node of the user to the root node. Thus, when no user is revoked, key authority just needs to publish a key update computed on the polynomial of the root node. When a subset of the users is revoked, key authority first finds the minimal set of nodes in the tree which contains an ancestor (or, the node itself) among all the leaf nodes corresponding to non-revoked users. Then, it publishes key updates on polynomials of the nodes in this set. We first address chosen-plaintext attack only, and later show how to extend the scheme to resist chosen-ciphertext attacks, as well.

Construction. We now specify the scheme $\mathcal{RIBE}[\mathcal{G}] = (\mathcal{S}, \mathcal{SK}, \mathcal{KU}, \mathcal{DK}, \mathcal{E}, \mathcal{D}, \mathcal{R})$, where $\mathcal{G}$ is the bilinear group generator. We assume that all users agree on how time is divided by time periods, and how each time period is specified, e.g., by days and "04.14.08". In our $\mathcal{RIBE}$ scheme, messages are encrypted using identity and time. Identity is a string associated with any user, e.g., an email "abc@xyz.com". Time indicates when the ciphertext is supposed to be decrypted, e.g., on 04.14.08. The message space MsgSp is $\mathbb{G}_T$. The identity space IdSp is $\{0,1\}^*$, and the time space TimeSp is an arbitrary bitstring set of size polynomial in the security parameter. We require that the strings specifying identities and times can be distinguished, e.g., by reserving the most significant bit (MSB): 0 for identity strings, and 1 for time strings. In our construction, identity and time strings are mapped to unique elements of $\mathbb{Z}_p^*$ (if needed, a collision-resistant hash function mapping $\{0,1\}^*$ to $\mathbb{Z}_p^*$ can be used). From now on for simplicity, we assume that identity and time are distinguished elements in $\mathbb{Z}_p^*$.

For any $x, i \in \mathbb{Z}$, $\mathsf{J} \subseteq \mathbb{Z}$, the *Lagrange coefficient* $\Delta_{i,\mathsf{J}}(x)$ is defined as

$$\Delta_{i,\mathsf{J}}(x) \stackrel{\text{def}}{=} \prod_{j \in \mathsf{J},\ j \neq i} \left(\frac{x-j}{i-j} \right).$$

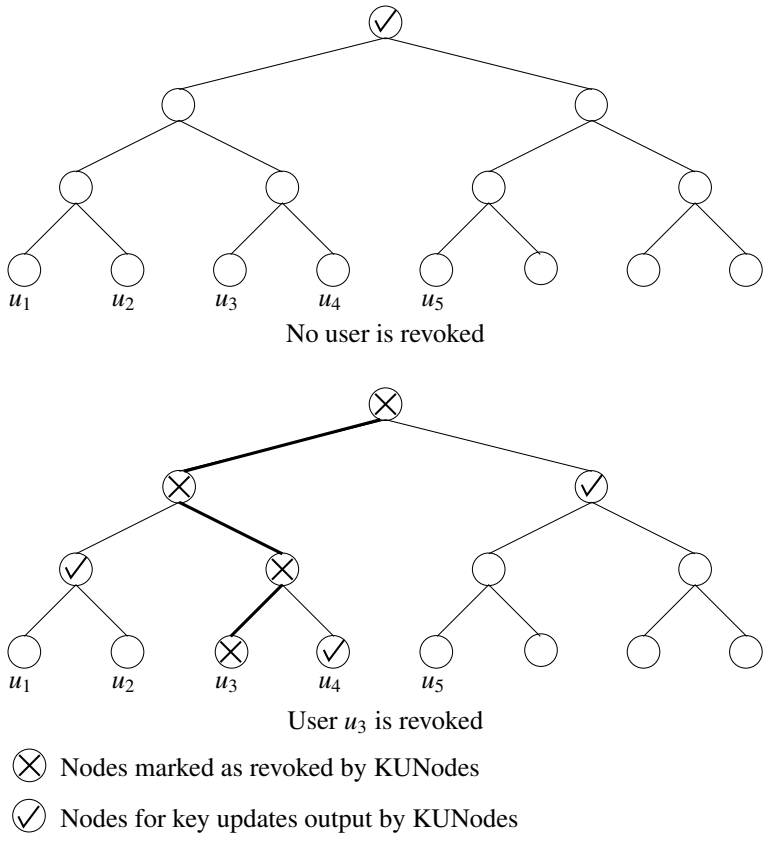

Figure 4: Pictorial Description of KUNodes Function

For any $x \in \mathbb{Z}, g \in \mathbb{G}_T, \mathsf{J} \subseteq \mathbb{Z}, h_1, \ldots, h_{|\mathsf{J}|} \in \mathbb{G}$, we define

$$H_{g,\mathsf{J},h_1,\ldots,h_{|\mathsf{J}|}}(x) \stackrel{\text{def}}{=} g^{x^2} \prod_{i=1}^{|\mathsf{J}|} \left(h_i^{\Delta_{i,\mathsf{J}}(x)}\right) .$$

Our construction uses the binary tree data structure, so we introduce some notation now. We denote by root, the root node. If v is a leaf node, then Path(v) denotes the set of nodes on the path from v to root (both v and root inclusive). If v is a non-leaf node, then v_l, v_r denote the left and the right child of v. We assume that nodes in the tree are uniquely encoded as strings, and the tree is defined by all of its nodes descriptions.

We also define a function KUNodes that is used to compute the minimal set of nodes for which key update needs to be published, so that only non-revoked users at time t are able to decrypt ciphertexts.[6] The function takes input a binary tree T, a revocation list *rl*

[6] A similar function was used in [2].

and a time t, and outputs a set of nodes, which is the minimal set of nodes in T, such that none of the nodes in rl with corresponding time $\leq t$ (users revoked on or before t) have any ancestor (or, themselves) in the set, and all other leaf nodes (corresponding to non-revoked users) have exactly one ancestor (or, themselves) in the set. The function operates as follows. First mark all the ancestors of revoked nodes as revoked, then output all the non-revoked children of revoked nodes. Refer to Figure 4 for a pictorial depiction. Here is a formal specification.

$\mathsf{KUNodes}(\mathsf{T}, rl, t)$

$\mathsf{X}, \mathsf{Y} \leftarrow \phi$

$\forall (v_i, t_i) \in rl$

if $t_i \leq t$, then add $\mathsf{Path}(v_i)$ to X

$\forall x \in \mathsf{X}$

if $x_l \notin \mathsf{X}$, then add x_l to Y

if $x_r \notin \mathsf{X}$, then add x_r to Y

if $\mathsf{Y} = \phi$, then add root to Y

return Y

We are now ready to present the description of Revocable IBE. We could not use the algorithms of the Fuzzy IBE construction from [83] in a black-box manner. The reason is that there the polynomial for each key is picked independently by the key generation algorithm. And in our construction some polynomials need to be shared by different keys. After we provide the details for each algorithm, we give some intuition and relation to the construction from [83] following "//" sign.

Construction 5.2.1. Let $\mathcal{G}$ be a prime order bilinear group generator. Let J be the set $\{1, 2, 3\}$.

- **Setup** $\mathcal{S}(1^\kappa, n)$: $(\tilde{\mathbb{G}}, p, g) \xleftarrow{\$} \mathcal{G}(1^\kappa)$; $a \xleftarrow{\$} \mathbb{Z}_p$; $g_1 \leftarrow g^a$; $g_2, h_1, h_2, h_3 \xleftarrow{\$} \mathbb{G}$. Let rl be an empty set, and T be a binary tree with at least n leaf nodes. Return $pk = (g, g_1, g_2, h_1, h_2, h_3)$; $mk = a$; rl ; $st = \mathsf{T}$.

// Besides the additional outputs of rl and st, it is essentially the same as Setup of Fuzzy IBE, in which 2 out of 2 attributes need to be matched.

- **Private Key Generation** $\mathcal{SK}(pk, mk, \omega, st)$: Parse pk as $(g, g_1, g_2, h_1, h_2, h_3)$, mk as a, st as T.[7] Pick an unassigned leaf node v from T, and store ω in that node.

  $\forall x \in \mathsf{Path}(v)$

  if a_x is undefined, then $a_x \xleftarrow{\$} \mathbb{Z}_p$, store a_x in node x

  $r_x \xleftarrow{\$} \mathbb{Z}_p$; $D_x \leftarrow g_2^{a_x\omega+a} H_{g_2,\mathsf{J},h_1,h_2,h_3}(\omega)^{r_x}$; $d_x \leftarrow g^{r_x}$

  Return $sk_\omega = \{(x, D_x, d_x)\}_{x \in \mathsf{Path}(v)}$; st.

  // We note that a_x above fixes a first-degree polynomial $q_x(y) = a_x y + a$ corresponding to node x. The algorithm computes the ω-components of the decryption key using the polynomials of all the nodes on the path from leaf node corresponding to ω to the root node.

- **Key Update Generation** $\mathcal{KU}(pk, mk, t, rl, st)$: Parse pk as $(g, g_1, g_2, h_1, h_2, h_3)$, mk as a, st as T.

  $\forall x \in \mathsf{KUNodes}(\mathsf{T}, rl, t)$

  $r_x \xleftarrow{\$} \mathbb{Z}_p$; $E_x \leftarrow g_2^{a_x t+a} H_{g_2,\mathsf{J},h_1,h_2,h_3}(t)^{r_x}$; $e_x \leftarrow g^{r_x}$

  Return $ku_t = \{(x, E_x, e_x)\}_{x \in \mathsf{KUNodes}(\mathsf{T},rl,t)}$.

  // The algorithm first finds a minimal set of nodes which contains an ancestor (or, the node itself) of all the non-revoked nodes. Then it computes the t- component of the decryption key using the polynomials of all the nodes in that set.

- **Decryption Key Generation** $\mathcal{DK}(sk_\omega, ku_t)$: Parse sk_ω as $\{(i, D_i, d_i)\}_{i \in \mathcal{I}}$, ku_t as $\{(j, E_j, e_j)\}_{j \in \mathsf{J}}$ for some set of nodes $\mathcal{I}, \mathsf{J}$.

[7] Every node x in T stores an element $a_x \in \mathbb{Z}_p$, and in addition, every leaf node stores an identity ω. If no such identity is stored at a leaf node, we say that the leaf node is unassigned.

$\forall (i, D_i, d_i) \in sk_\omega, (j, E_j, e_j) \in ku_t$

if $\exists (i, j)$ s.t. $i = j$, then $dk_{\omega,t} \leftarrow (D_i, E_j, d_i, e_j)$

else if sk_ω and ku_t don't have any node in common, then $dk_{\omega,t} \leftarrow \perp$

Return $dk_{\omega,t}$.

// Above we can drop the subscripts i, j since they are equal, i.e., $dk_{\omega,t} = (D, E, d, e)$. The algorithm finds components of sk_ω and ku_t which were computed on the same polynomial.

- **Encryption** $\mathcal{E}(pk, \omega, t, m)$: Parse pk as $(g, g_1, g_2, h_1, h_2, h_3)$. $z \xleftarrow{\$} \mathbb{Z}_p$; $c_1 \leftarrow m \cdot \mathbf{e}(g_1, g_2)^z$; $c_2 \leftarrow g^z$; $c_\omega \leftarrow H_{g_2,\mathsf{J},h_1,h_2,h_3}(\omega)^z$; $c_t \leftarrow H_{g_2,\mathsf{J},h_1,h_2,h_3}(t)^z$. Return $c = (\omega, t, c_\omega, c_t, c_1, c_2)$.

  // The Encryption algorithm is essentially the same as that of Fuzzy IBE.

- **Decryption** $\mathcal{D}(dk_{\omega,t}, c)$: Parse $dk_{\omega,t}$ as (D, E, d, e), c as $(\omega, t, c_\omega, c_t, c_1, c_2)$. $m \leftarrow c_1 \left(\frac{\mathbf{e}(d,c_\omega)}{\mathbf{e}(D,c_2)}\right)^{\frac{t}{t-\omega}} \left(\frac{\mathbf{e}(e,c_t)}{\mathbf{e}(E,c_2)}\right)^{\frac{\omega}{\omega-t}}$. Return m.

  // The decryption algorithm is essentially the same as that of Fuzzy IBE.

- **Revocation** $\mathcal{R}(\omega, t, rl, st)$: For all nodes v associated with identity ω, add (v, t) to rl. Return rl.

Consistency. If identity ω was not revoked before, or at time t, then we will show that $\mathcal{D}(dk_{\omega,t}, c) = m$, where $dk_{\omega,t}, m$ and c are computed as per the consistency requirement in Section 5.1.1.

From the definition of KUNodes we see that if ω was not revoked before or, at t then the set of nodes output by KUNodes has one ancestor (or, the node itself) of the leaf node associated with ω, which implies that there will be a common node in sk_ω, and ku_t and hence $\mathcal{DK}$ will not output $\perp$. Now from the above construction we have that for $a, a_x, z, r_\omega, r_t \in \mathbb{Z}_p$: $g, g_2, h_1, h_2, h_3 \in \mathbb{G}$; $g_1 = g^a$; $dk_{\omega,t} = (D, E, d, e)$, where $D =$

$g_2^{a_x\omega+a} H_{g_2,\mathsf{J},h_1,h_2,h_3}(\omega)^{r_\omega}$, $E = g_2^{a_x t+a} H_{g_2,\mathsf{J},h_1,h_2,h_3}(t)^{r_t}$, $d = g^{r_\omega}$, $e = g^{r_t}$; $c = (\omega, t, c_\omega, c_t, c_1, c_2)$, where $c_\omega = H_{g_2,\mathsf{J},h_1,h_2,h_3}(\omega)^z$, $c_t = H_{g_2,\mathsf{J},h_1,h_2,h_3}(t)^z$, $c_1 = m \cdot \mathbf{e}(g_1, g_2)^z$, $c_2 = g^z$.

So, $\mathcal{D}(dk_{\omega,t}, c)$

$$
\begin{aligned}
&= c_1 \left(\frac{\mathbf{e}(d, c_\omega)}{\mathbf{e}(D, c_2)}\right)^{\frac{t}{t-\omega}} \left(\frac{\mathbf{e}(e, c_t)}{\mathbf{e}(E, c_2)}\right)^{\frac{\omega}{\omega-t}} \\
&= m \cdot \mathbf{e}(g_1, g_2)^z \cdot \left(\frac{\mathbf{e}(g^{r_\omega}, H_{g_2,\mathsf{J},h_1,h_2,h_3}(\omega)^z)}{\mathbf{e}(g_2^{a_x\omega+a} H_{g_2,\mathsf{J},h_1,h_2,h_3}(\omega)^{r_\omega}, g^z)}\right)^{\frac{t}{t-\omega}} \cdot \left(\frac{\mathbf{e}(g^{r_t}, H_{g_2,\mathsf{J},h_1,h_2,h_3}(t)^z)}{\mathbf{e}(g_2^{a_x t+a} H_{g_2,\mathsf{J},h_1,h_2,h_3}(t)^{r_t}, g^z)}\right)^{\frac{\omega}{\omega-t}} \\
&= m \cdot \mathbf{e}(g_1, g_2)^z \left(\frac{1}{\mathbf{e}(g_2^{a_x\omega+a}, g^z)}\right)^{\frac{t}{t-\omega}} \cdot \left(\frac{1}{\mathbf{e}(g_2^{a_x t+a}, g^z)}\right)^{\frac{\omega}{\omega-t}} \\
&= m \cdot \mathbf{e}(g_1, g_2)^z \cdot \left(\frac{1}{\mathbf{e}(g_2^{(a_x\omega+a)(\frac{t}{t-\omega})+(a_x t+a)(\frac{\omega}{\omega-t})}, g^z)}\right) \\
&= m \cdot \mathbf{e}(g_1, g_2)^z \frac{1}{\mathbf{e}(g_2^a, g^z)} \\
&= m \cdot \mathbf{e}(g_1, g_2)^z \frac{1}{\mathbf{e}(g_2, g_1)^z} \\
&= m.
\end{aligned}
$$

Remarks. The function KUNodes needs to be executed only when *rl* has changed, so key authority can store the output of KUNodes, and use it until *rl* changes. If the number of users exceeds n, the capacity of the current tree, it is possible to extend the tree and permit n more users as follows. Take an "empty" tree of the same size and connect the roots of the current and new trees to the new parent root node. Now the combined tree has $2n$ leaf nodes, and new users can be accommodated. Each user will need an additional private key component computed on the polynomial of the new root node. This new private key component can be encrypted (under the corresponding identity and time), and published.

Efficiency. We first analyze communication and time complexity of key authority in computing and publishing key updates as a function of the number of users n and number of revoked users r. We compare *the worst case* complexity of our scheme with that of the general revocation solution suggested by Boneh-Franklin [34] that we outlined in the Introduction. Table 1 summarizes the results. The complexity analysis for our construction

follows directly from Theorem 1 of [2], as the number of necessary key updates in our scheme corresponds to the number of nodes returned by function KUNodes, and a similar function on the binary tree was used in [2].

As the table shows, our scheme represents a significant improvement over the Boneh-Franklin solution for small values of r. For larger values of r (especially as it reaches close to n), this advantage is lost. We however note that as r becomes large, our scheme can be "reset" to keep key update efficient (by running the setup algorithm again which will make the revocation list empty and releasing new private keys for only non-revoked users).

In terms of encryption and decryption, our construction is slightly less efficient than the existing IBE schemes. E.g., the decryption algorithms of IBEs by Waters [88] and Boneh-Boyen [31] require 2 pairing computations (the slowest computation compared to group operations and exponentiations), and our scheme requires 4. Encryption in the schemes of [88, 31] is dominated by 3 and 4 exponentiations, while our scheme uses 12. We chose Waters and Boneh-Boyen constructions for comparison, because they are the most efficient IBE schemes secure in standard (RO devoid) model under standard assumptions. This may be a reasonable price to pay for the significant improvement in key-update efficiency, which may become a bottleneck for a large number of users. We note that the size of secret keys is larger in our scheme, a user needs to store up to $3h = 3 \log n$ group elements.

We note that using the suggestion from [80], efficiency of our scheme, and in particular, its encryption algorithm, can be improved, if a hash function is used in place of the function H. Security analysis in this case will need to rely on the random oracle (RO) model [18]. This will improve the number of exponentiations in encryption to 4, while the decryption algorithm will still be dominated by 4 paring operations. In contrast, the cost of encryption and decryption in the Boneh-Franklin scheme [34] is dominated by one pairing each.

Security. Even though different users have their private keys computed on the same polynomial, this does not introduce insecurity in $\mathcal{RIBE}$ as opposed to Fuzzy IBE. In our scheme, collusion among different users is possible, however such collusion is not useful. No matter

Table 1: Key Update Complexity (# of users n, # of revoked users r)

	$r = 0$	$1 < r \leq n/2$	$n/2 < r \leq n$
BF [34]	$O(n)$	$O(n - r)$	$O(n - r)$
Revocable IBE	$O(1)$	$O(r \log(\frac{n}{r}))$	$O(n - r)$

how many revoked users try to collude, they will still be unable to decrypt a ciphertext for a new time period, as they cannot obtain the necessary decryption key component. One might be tempted to reduce the security of $\mathcal{RIBE}$ to the security of Fuzzy IBE, since after all $\mathcal{RIBE}$ uses Fuzzy IBE as its base construction. However, we want to point out that such a straightforward reduction of security in a black box manner does not seem possible. The main reason for this is that in Fuzzy IBE, each time key generation algorithm is run, it chooses a random polynomial, and then computes the key using that polynomial. However, in $\mathcal{RIBE}$ it is essential that private key and key update be computed on some fixed polynomials.

We now state the security result.

Theorem 5.2.2. Let $\mathcal{G}$ be a prime order bilinear group generator, and $\mathcal{RIBE}[\mathcal{G}] = (\mathcal{S}, \mathcal{SK}, \mathcal{KU}, \mathcal{DK}, \mathcal{E}, \mathcal{D}, \mathcal{R})$ be the associated Revocable IBE scheme, defined by Construction 5.2.1. Then for any adversary A attacking sRID-CPA security of $\mathcal{RIBE}$ with n users, whose running time is t_A, and who asks q_p private key generation queries, q_k key update generation queries and q_r revocation queries, there exists an adversary B solving DBDH problem for $\mathcal{G}$, such that

$$\mathbf{Adv}^{\text{srid-cpa}}_{\mathcal{RIBE},n}(A) \leq 4 \cdot \mathbf{Adv}^{\text{dbdh}}_{\mathcal{G}}(B) \,, \tag{12}$$

where the running time of B, $t_B = t_A + O(\kappa^3)$.

Proof of Theorem 5.2.2. We construct an adversary B for the DBDH problem associated with $\mathcal{G}$. B gets $(1^\kappa, \tilde{\mathbb{G}}, p, g, X, Y, Z, W)$ as input and it has to return a bit d. It is going to use A.

$B(1^\kappa, \tilde{\mathbb{G}}, p, g, X, Y, Z, W)$:

$(\omega^*, t^*, state) \xleftarrow{\$} A(1^\kappa)$

$g_1 \leftarrow X,\ g_2 \leftarrow Y$

Pick random second-degree polynomials $f(x), u(x)$ with coefficients in $\mathbb{Z}_p$,

s.t. $u(x) = -x^2$ for $x = \omega^*, t^*$, o.w. $u(x) \neq -x^2$

For $i = 1, 2, 3 : h_i \leftarrow g_2^{u(i)} g^{f(i)}$

$pk \leftarrow (g, g_1, g_2, h_1, h_2, h_3)$

Let rl be an empty set, and T be a binary tree with at least n leaf nodes.

Pick a leaf node v^* from T, and a random bit rev

Run A, and answer its queries to the $\mathcal{SK}(\cdot), \mathcal{KU}(\cdot), \mathcal{R}(\cdot,\cdot)$ oracles using the subroutines defined.

$(m_0, m_1, state') \xleftarrow{\$} A^{\mathcal{SK}(\cdot),\mathcal{KU}(\cdot),\mathcal{R}(\cdot,\cdot)}(1^\kappa, pk, state)$

$b \xleftarrow{\$} \{0, 1\}$

$c_1^* \leftarrow m_b \cdot W,\ c_2^* \leftarrow Z,\ c_\omega^* \leftarrow Z^{f(\omega^*)},\ c_t^* \leftarrow Z^{f(t^*)}$

$c^* \leftarrow (\omega^*, t^*, c_\omega^*, c_t^*, c_1^*, c_2^*)$

$d \xleftarrow{\$} A^{\mathcal{SK}(\cdot),\mathcal{KU}(\cdot),\mathcal{R}(\cdot,\cdot)}(1^\kappa, pk, c^*, state')$

if any of the oracles abort, return 1

else if $b = d$ return 1 else, return 0

Now, we present the subroutines for answering the oracle queries. These subroutines use the functions that we define here. For $i, j, l, r' \in \mathbb{Z}_p,\ S = \{0, j\}$ define

$$F_1(i, j, l, r') \stackrel{\text{def}}{=} g_2^{l\Delta_{j,S}(i)} \left(g_1^{\frac{-f(i)}{i^2+u(i)}} \left(g_2^{i^2+u(i)} g^{f(i)} \right)^{r'} \right)^{\Delta_{0,S}(i)},$$

$$F_2(i, r') \stackrel{\text{def}}{=} \left(g_1^{\frac{-1}{i^2+u(i)}} g^{r'} \right)^{\Delta_{0,S}(i)}.$$

$\mathcal{SK}(\omega)$:

if $rev = 0$ and $\omega = \omega^*$, then abort

if $rev = 1$ and $\omega = \omega^*$, then:

$v \leftarrow v^*$,

$\forall x \in \mathsf{Path}(v)\ r_x \xleftarrow{\$} \mathbb{Z}_p$

if $\nexists l_x$, then $l_x \xleftarrow{\$} \mathbb{Z}_p$, and store l_x in node x

$D_x \leftarrow g_2^{l_x} H_{g_2,h_1,h_2,h_3}(\omega^*)^{r_x},\ d_x \leftarrow g^{r_x}$

if $rev = 0$ and $\omega \neq \omega^*$, then:

pick an unassigned leaf node v from T, and store ω in node v

$\forall x \in \mathsf{Path}(v)\ r'_x \xleftarrow{\$} \mathbb{Z}_p$,

if $\nexists l_x$, then $l_x \xleftarrow{\$} \mathbb{Z}_p$, and store l_x in node x

$i \leftarrow \omega, j \leftarrow t^*, l \leftarrow l_x, r' \leftarrow r'_x$

$D_x \leftarrow F_1(i, j, l, r'),\ d_x \leftarrow F_2(i, r')$

if $rev = 1$ and $\omega \neq \omega^*$, then:

pick an unassigned leaf node v from T, and store ω in node v

$\forall x \in \mathsf{Path}(v)\ r'_x \xleftarrow{\$} \mathbb{Z}_p$,

if $\nexists l_x$, then $l_x \xleftarrow{\$} \mathbb{Z}_p$, and store l_x in node x

$i \leftarrow \omega, l \leftarrow l_x, r' \leftarrow r'_x$

$\forall x \in (\mathsf{Path}(v) \setminus \mathsf{Path}(v^*))$

$j \leftarrow t^*$

$D_x \leftarrow F_1(i, j, l, r'),\ d_x \leftarrow F_2(i, r')$

$\forall x \in (\mathsf{Path}(v) \cap \mathsf{Path}(v^*))$

$j \leftarrow \omega^*$

$D_x \leftarrow F_1(i, j, l, r'),\ d_x \leftarrow F_2(i, r')$

return $sk_\omega = \{(x, D_x, d_x)\}_{x \in \mathsf{Path}(v)}$

$\mathcal{R}(\omega, t)$:

for all leaf nodes v associated with identity ω, add (v, t) to rl

$\mathcal{KU}(t)$:

if $rev = 1$ and $t = t^*$ and $\forall t \leq t^*$ we have that $(\omega^*, t) \notin rl$, then abort

else if $t = t^*$, then:

$\forall x \in \mathsf{KUNodes}(\mathsf{T}, rl, t), r_x \xleftarrow{\$} \mathbb{Z}_p$

$E_x \leftarrow g_2^{l_x} H_{g_2,h_1,h_2,h_3}(t^*)^{r_x},\ e_x \leftarrow g^{r_x}$

if $rev = 1$ and $t \neq t^*$, then:

$\forall x \in (\mathsf{KUNodes}(\mathsf{T}, rl, t) \setminus \mathsf{Path}(v^*))$

$r'_x \xleftarrow{\$} \mathbb{Z}_p, i \leftarrow t, j \leftarrow t^*, l \leftarrow l_x, r' \leftarrow r'_x$

$E_x \leftarrow F_1(i, j, l, r'),\ e_x \leftarrow F_2(i, r')$

$\forall x \in (\mathsf{KUNodes}(\mathsf{T}, rl, t) \cap \mathsf{Path}(v^*))$

$r'_x \xleftarrow{\$} \mathbb{Z}_p, i \leftarrow t, j \leftarrow \omega^*, l \leftarrow l_x, r' \leftarrow r'_x$

$E_x \leftarrow F_1(i, j, l, r'),\ e_x \leftarrow F_2(i, r')$

if $rev = 0$ and $t \neq t^*$, then:

$\forall x \in \mathsf{KUNodes}(\mathsf{T}, rl, t)$

$r'_x \xleftarrow{\$} \mathbb{Z}_p, i \leftarrow t, j \leftarrow t^*, l \leftarrow l_x, r' \leftarrow r'_x$

$E_x \leftarrow F_1(i, j, l, r'),\ e_x \leftarrow F_2(i, r')$

return $ku_t = \{(x, E_x, e_x)\}_{x \in \mathsf{KUNodes}(\mathsf{T}, rl, t)}$

Analysis. $f(x)$ being a random polynomial ensures that h_1, h_2, h_3 are random, so pk has the right distribution. For $i = 1, 2, 3$ $h_i = g_2^{u(i)} g^{f(i)}$, which implies $H_{g_2,h_1,h_2,h_3}(x) = g_2^{x^2+u(x)} g^{f(x)}$. Since $x = \omega^*, t^*$ $u(x) = -x^2$, then $H_{g_2,h_1,h_2,h_3}(\omega^*) = g^{f(\omega^*)}, H_{g_2,h_1,h_2,h_3}(t^*) = g^{f(t^*)}$. If $W = \mathbf{e}(g, g)^{xyz}$, then $c_1^* = m_b \cdot W = m_b \cdot \mathbf{e}(g_1, g_2)^z, c_{\omega^*} = Z^{f(\omega^*)} = g^{zf(\omega^*)} = H_{g_2,h_1,h_2,h_3}(\omega^*)^z, c_{t^*} = Z^{f(t^*)} = g^{zf(t^*)} = H_{g_2,h_1,h_2,h_3}(t^*)^z$. So, if B is in $\mathbf{Exp}_{\mathcal{G}}^{\text{dbdh-real}}(B)$, then c^* is a well-formed

ciphertext of m_b. Otherwise, if $W = \mathbf{e}(g,g)^w$ for a random w, then $c_1^* = m_b \cdot W = m_b \cdot \mathbf{e}(g,g)^w$, so w being random and independent from x, y, z ensures that c_1^* is also random, and thus c^* hides bit b information-theoretically. In what follows, let $g_1 = g^a$ (to simplify notation).

Case 1. rev $= 1, \omega = \omega^*$ in $\mathcal{SK}(\omega)$ oracle simulation:

Define $a_x = \frac{1}{\omega^*}(l_x - a)$. Then $D_x = g_2^{a_x\omega^*+a} H_{g_2,h_1,h_2,h_3}(\omega^*)^{r_x}, d_x = g^{r_x}$. So, sk_ω has the right distribution.

Case 2. rev $= 0, \omega \neq \omega^*$ in $\mathcal{SK}(\omega)$ oracle simulation:

Define $a_x = \frac{1}{t^*}(l_x - a)$, $r_x = (r'_x - \frac{a}{\omega^2+u(\omega)})\Delta_{0,S}(\omega)$. Then

$$\begin{aligned} F_2(i,r') &= \left(g_1^{\frac{-1}{i^2+u(i)}} g^{r'}\right)^{\Delta_{0,S}(i)} \\ &= \left(g^{r'-\frac{a}{i^2+u(i)}}\right)^{\Delta_{0,S}(i)} \\ &= \left(g^{r'_x-\frac{a}{\omega^2+u(\omega)}}\right)^{\Delta_{0,S}(\omega)} \\ &= g^{r'_x} \end{aligned}$$

and

$$\begin{aligned} F_1(i,j,l,r') &= g_2^{l\Delta_{j,S}(i)} \left(\left(g_1^{\frac{-f(i)}{i^2+u(i)}}\right)\left(g_2^{i^2+u(i)} g^{f(i)}\right)^{r'}\right)^{\Delta_{0,S}(i)} \\ &= g_2^{l\Delta_{j,S}(i)} \left(\left(g^{\frac{-af(i)}{i^2+u(i)}}\right)\left(g_2^{i^2+u(i)} g^{f(i)}\right)^{r'}\right)^{\Delta_{0,S}(i)} \\ &= g_2^{l\Delta_{j,S}(i)} \left(g_2^a \left(g_2^{i^2+u(i)} g^{f(i)}\right)^{\frac{-a}{i^2+u(i)}} \left(g_2^{i^2+u(i)} g^{f(i)}\right)^{r'}\right)^{\Delta_{0,S}(i)} \\ &= g_2^{l\Delta_{j,S}(i)} \left(g_2^a \left(g_2^{i^2+u(i)} g^{f(i)}\right)^{r'-\frac{a}{i^2+u(i)}}\right)^{\Delta_{0,S}(i)} \\ &= g_2^{l\Delta_{j,S}(i)+a\Delta_{0,S}(i)} \left(g_2^{i^2+u(i)} g^{f(i)}\right)^{\left(r'-\frac{a}{i^2+u(i)}\right)\Delta_{0,S}(i)} \\ &= g_2^{l\Delta_{j,S}(i)+a\Delta_{0,S}(i)} H_{g_2,h_1,h_2,h_3}(i)^{\left(r'-\frac{a}{i^2+u(i)}\right)\Delta_{0,S}(i)} \\ &= g_2^{l_x\Delta_{j,S}(\omega)+a\Delta_{0,S}(\omega)} H_{g_2,h_1,h_2,h_3}(\omega)^{\left(r'_x-\frac{a}{\omega^2+u(\omega)}\right)\Delta_{0,S}(\omega)} \\ &= g_2^{a_x\omega+a} H_{g_2,h_1,h_2,h_3}(\omega)^{r_x}. \end{aligned}$$

So, sk_ω has the right distribution.

Case 3. rev $= 1$ and $\omega \neq \omega^*$. Similar arguments as above apply if we define

$a_x = \frac{1}{\omega^*}(l_x - a),\ r_x = (r'_x - \frac{a}{\omega^2+u(\omega)})\Delta_{0,S}(\omega)$ for nodes on the path of v^* and

$a_x = \frac{1}{t^*}(l_x - a),\ r_x = (r'_x - \frac{a}{\omega^2+u(\omega)})\Delta_{0,S}(\omega)$ for rest of the nodes.

Similar arguments as above apply for all cases in $\mathcal{KU}(t)$ oracle simulation if we define:

In case $rev = 0$: $a_x = \frac{1}{t^*}(l_x - a),\ r_x = (r'_x - \frac{a}{t^2+u(t)})\Delta_{0,S}(t)$ for all nodes.

And, in case $rev = 1$: $a_x = \frac{1}{\omega^*}(l_x - a),\ r_x = (r'_x - \frac{a}{t^2+u(t)})\Delta_{0,S}(t)$ for nodes on the path of v^*, and $a_x = \frac{1}{t^*}(l_x - a),\ r_x = (r'_x - \frac{a}{t^2+u(t)})\Delta_{0,S}(t)$ for rest of the nodes.

Note that we are defining a_x consistently in both oracle simulations, i.e., in case $rev = 0$, $a_x = \frac{1}{t^*}(l_x - a)$ for all nodes, and in case $rev = 1$, $a_x = \frac{1}{\omega^*}(l_x - a)$ for all nodes on the path from v^* to the root node, $a_x = \frac{1}{t^*}(l_x - a)$ for rest of the nodes. Also note that values l_x which are stored in node x, and identities ω which are stored in leaf nodes of tree T, have the right distribution, and also that they are modified only by $\mathcal{SK}(\omega)$ oracle; $\mathcal{KU}(t)$ and $\mathcal{R}(\omega, t)$ oracles do not modify them. Similarly, revocation list rl is modified only by $\mathcal{R}(\omega, t)$ oracle; $\mathcal{SK}(\omega)$ and $\mathcal{KU}(t)$ oracles do not modify them. Thus, oracles maintain the state and revocation list consistently and with right distribution.

Claim 5.2.3. Let $\mathsf{sreal}, \mathsf{srand}$ denote the events that none of the oracles abort in $\mathbf{Exp}^{\text{dbdh-real}}_{\mathcal{G}}(B)$, $\mathbf{Exp}^{\text{dbdh-rand}}_{\mathcal{G}}(B)$ respectively. Then,

$$\Pr[\ \mathsf{sreal}\ ] = \Pr[\ \mathsf{srand}\ ] \geq \frac{1}{2}.$$

Proof of Claim 5.2.3. We will prove the claim in two parts. First we will show that $\Pr[\ \mathsf{sreal}\ ] = \Pr[\ \mathsf{srand}\ ]$ and then we will show that $\Pr[\ \mathsf{sreal}\ ] \geq \frac{1}{2}$.

First part is easy to see. The probability that $\mathcal{SK}(\omega)$ and $\mathcal{KU}(t)$ oracles abort depends on the bit rev which is chosen independently from whether B is in $\mathbf{Exp}^{\text{dbdh-real}}_{\mathcal{G},B}(1^\kappa)$ or $\mathbf{Exp}^{\text{dbdh-rand}}_{\mathcal{G},B}(1^\kappa)$. So, $\Pr[\ \mathsf{sreal}\ ] = \Pr[\ \mathsf{srand}\ ]$. Now it remains to show that $\Pr[\ \mathsf{sreal}\ ] \geq \frac{1}{2}$.

Condition 3 of Definition 5.1.2 says that $\mathcal{SK}(\omega)$ oracle can be queried on ω^* only if

$\mathcal{R}(\omega, t)$ oracle was queried on (ω^*, t) for any $t \leq t^*$. Thus, we have

$$\begin{aligned}
& \Pr[\ \omega = \omega^*\ ] \leq \Pr[\ (\omega^*, t) \in rl,\ \forall\ t \leq t^*\ ] \\
\Rightarrow\quad & 1 - \Pr[\ \omega = \omega^*\ ] \geq \Pr[\ (\omega^*, t) \notin rl,\ \forall\ t \leq t^*\ ] \\
\Rightarrow\quad & 1 - \Pr[\ \omega = \omega^*\ ] \geq \Pr[\ (t = t^*) \wedge ((\omega^*, t) \notin rl,\ \forall\ t \leq t^*)\ ].
\end{aligned}$$

We see that $\mathcal{SK}(\omega)$ oracle aborts, if $rev = 0$, and $\omega = \omega^*$ and $\mathcal{KU}(t)$ oracle aborts if $rev = 1, t = t^*$ and $\forall t \leq t^*\ (\omega^*, t) \notin rl$. Thus,

$$\begin{aligned}
\Pr\left[\ \overline{\mathsf{sreal}}\ \right] &= \Pr[\ (rev = 0) \wedge (\omega = \omega^*)\ ] \\
&\quad + \Pr[\ (rev = 1) \wedge (t = t^*) \wedge ((\omega^*, t) \notin rl,\ \forall\ t \leq t^*)\ ] \\
&= \Pr[\ rev = 0\ ] \cdot \Pr[\ \omega = \omega^*\ ] \\
&\quad + \Pr[\ rev = 1\ ] \cdot \Pr[\ (t = t^*) \wedge ((\omega^*, t) \notin rl,\ \forall\ t \leq t^*)\ ] \\
&\leq \frac{1}{2} \Pr[\ \omega = \omega^*\ ] + \frac{1}{2}(1 - \Pr[\ \omega = \omega^*\ ]) \\
&\leq \frac{1}{2}.
\end{aligned}$$

Therefore, $\Pr[\ \mathsf{sreal}\ ] \geq \frac{1}{2}$. □

We have shown above that when B is in $\mathbf{Exp}_{\mathcal{G}}^{\text{dbdh-real}}(B)$, and none of the oracles abort, then B is simulating the exact experiment $\mathbf{Exp}_{\mathcal{RIBE},n}^{\text{srid-cpa}}(A)$ for A. So,

$$\Pr\left[\ \mathbf{Exp}_{\mathcal{G}}^{\text{dbdh-real}}(B) = 1 \mid \mathsf{sreal}\ \right] \geq \Pr\left[\ \mathbf{Exp}_{\mathcal{RIBE},n}^{\text{srid-cpa}}(A) = 1\ \right].$$

When B is in $\mathbf{Exp}_{\mathcal{G}}^{\text{dbdh-rand}}(B)$, and none of the oracles abort then as explained earlier bit b is information-theoretically hidden from A. So,

$$\Pr\left[\ \mathbf{Exp}_{\mathcal{G}}^{\text{dbdh-rand}}(B) = 1 \mid \mathsf{srand}\ \right] \leq \frac{1}{2}.$$

Also, since B outputs 1, if either of the oracles aborts, so

$$\begin{aligned}
&\Pr\left[\ \mathbf{Exp}_{\mathcal{G}}^{\text{dbdh-real}}(B) = 1 \mid \overline{\mathsf{sreal}}\ \right] = 1, \\
&\Pr\left[\ \mathbf{Exp}_{\mathcal{G}}^{\text{dbdh-rand}}(B) = 1 \mid \overline{\mathsf{srand}}\ \right] = 1
\end{aligned}$$

Thus,

$$\begin{aligned}
\mathbf{Adv}^{\text{dbdh}}_{\mathcal{G}}(B) &= \Pr\left[\, \mathbf{Exp}^{\text{dbdh-real}}_{\mathcal{G}}(B) = 1 \,\right] - \Pr\left[\, \mathbf{Exp}^{\text{dbdh-rand}}_{\mathcal{G}}(B) = 1 \,\right] \\
&= \Pr[\,\mathsf{sreal}\,] \cdot \Pr\left[\, \mathbf{Exp}^{\text{dbdh-real}}_{\mathcal{G}}(B) = 1 | \mathsf{sreal} \,\right] \\
&\quad + \Pr\left[\, \overline{\mathsf{sreal}} \,\right] \cdot \Pr\left[\, \mathbf{Exp}^{\text{dbdh-real}}_{\mathcal{G}}(B) = 1 | \overline{\mathsf{sreal}} \,\right] \\
&\quad - \Pr[\,\mathsf{srand}\,] \cdot \Pr\left[\, \mathbf{Exp}^{\text{dbdh-rand}}_{\mathcal{G}}(B) = 1 | \mathsf{srand} \,\right] \\
&\quad - \Pr\left[\, \overline{\mathsf{srand}} \,\right] \cdot \Pr\left[\, \mathbf{Exp}^{\text{dbdh-rand}}_{\mathcal{G}}(B) = 1 | \overline{\mathsf{srand}} \,\right] \\
&\geq \frac{1}{2} \cdot \left(\Pr\left[\, \mathbf{Exp}^{\text{dbdh-real}}_{\mathcal{G}}(B) = 1 | \mathsf{sreal} \,\right] - \Pr\left[\, \mathbf{Exp}^{\text{dbdh-rand}}_{\mathcal{G}}(B) = 1 | \mathsf{srand} \,\right] \right) \\
&\geq \frac{1}{2} \cdot \left(\Pr\left[\, \mathbf{Exp}^{\text{srid-cpa}}_{\mathcal{RIBE},n}(A) = 1 \,\right] - \frac{1}{2} \right) \\
&\geq \frac{1}{4} \cdot \mathbf{Adv}^{\text{srid-cpa}}_{\mathcal{RIBE},n}(A) \; .
\end{aligned}$$

Finally, we observe that $t_B = t_A + O(\kappa^3)$ as B does only a constant number of modulo exponentiations and multiplications and picks a constant number of random group elements (recall that by convention t_A includes the time of the experiment including computations done by the oracles. □

5.3 Addressing CCA Security

We suggest two ways to construct RIBE schemes that resist chosen-ciphertext attacks. Our first solution is a modification of our main construction. Our second solution is generic, in that it is based on any sRID-CPA secure scheme, though CCA security relies on the RO model.

$RIBE_{CCA}$ Construction. We combine the ideas of [32] (used there for a different problem of constructing an IND-CCA public-key encryption scheme) with the error-tolerance property of Fuzzy IBE to modify our Revocable IBE scheme. Changes are mainly in the encryption and decryption algorithms. We employ a strongly-unforgeable one-time signature scheme (cf. [32] that recalls the primitive, and its security definition). The setup algorithm of the new scheme is very similar to the one in Fuzzy IBE, where 2 out of 3 attributes of

ciphertexts should match with those of the decryption key. The private key generation and key update generation algorithms are very similar to those of $\mathcal{RIBE}$, except that we now use second-degree polynomials as opposed to first-degree polynomials in $\mathcal{RIBE}$. The encryption algorithm runs the key generation algorithm of one-time signature to obtain a signing key and a verification key, and then encrypts the message with three attributes: identity, time and verification key. Then it signs the resulting intermediate ciphertext using the signing key. The decryption algorithm verifies the signature, and that the ciphertext is properly formed (by using a ciphertext sanity check due to [52]) before decrypting.

Let $\mathcal{G}$ be a bilinear group generator and $\mathcal{OTS} = (\mathsf{SGen}, \mathcal{T}, \mathsf{Ver})$ be a strongly-unforgeable one-time signature scheme. Let $\mathcal{RIBE}[\mathcal{G}] = (\mathcal{S}, \mathcal{SK}, \mathcal{KU}, \mathcal{DK}, \mathcal{E}, \mathcal{D}, \mathcal{R})$ be the scheme of Construction 5.2.1. We define $\mathcal{RIBE}_{CCA}[\mathcal{G}, \mathcal{OTS}] = (\mathcal{S}', \mathcal{SK}', \mathcal{KU}', \mathcal{DK}, \mathcal{E}', \mathcal{D}', \mathcal{R})$ by specifying the differences from $\mathcal{RIBE}$. Here we require that identities, time periods, and the verification keys for the one-time signature output by SGen are mapped to distinguished elements in $\mathbb{Z}_p^*$ (e.g., by pre-pending "00", "01" and "11" to strings of these types, and then using a collision-resistant hash function that maps $\{0,1\}^*$ to $\mathbb{Z}_p^*$. Let J be $\{1,2,3,4\}$.

- **Setup** $\mathcal{S}'(\kappa, n)$: Everything is the same as in $\mathcal{S}$, except that pk has an additional element $h_4 \xleftarrow{\$} \mathbb{G}$, and thus $pk = (g, g_1, g_2, h_1, h_2, h_3, h_4)$.

- **Private Key Generation** $\mathcal{SK}'(pk, mk, \omega, st)$: Everything is the same as in $\mathcal{SK}$, except that now we pick a random second-degree polynomial $q_x(y)$ with coefficients in $\mathbb{Z}_p$, and the same restriction that $q_x(0) = a$. Parse pk as $(g, g_1, g_2, h_1, h_2, h_3, h_4)$, mk as a, st as T. Pick an unassigned leaf node v from T, and store ω in that node.

  $\forall x \in \mathsf{Path}(v)$

  if q_x is undefined, then pick a random second-degree polynomial q_x,

  s.t. $q_x(0) = a$, and store q_x in node x

  $r_x \xleftarrow{\$} \mathbb{Z}_p$; $D_x \leftarrow g_2^{q_x(\omega)} H_{g_2,\mathsf{J},h_1,h_2,h_3,h_4}(\omega)^{r_x}$; $d_x \leftarrow g^{r_x}$

Return $sk_\omega = \{(x, D_x, d_x)\}_{x \in \mathsf{Path}(v)}$; st.

- **Key Update Generation** $\mathcal{KU}'(pk, mk, t, rl, st)$: Parse pk as $(g, g_1, g_2, h_1, h_2, h_3, h_4)$, mk as a, st as T.

  $\forall x \in \mathsf{KUNodes}(\mathsf{T}, rl, t)$

  $r_x \xleftarrow{\$} \mathbb{Z}_p$; $E_x \leftarrow g_2^{q_x(t)} H_{g_2,\mathsf{J},h_1,h_2,h_3,h_4}(t)^{r_x}$; $e_x \leftarrow g^{r_x}$

  Return $ku_t = \{(x, E_x, e_x)\}_{x \in \mathsf{KUNodes}(\mathsf{T},rl,t)}$.

- **Encryption** $\mathcal{E}'(pk, \omega, t, m)$: Parse pk as $(g, g_1, g_2, h_1, h_2, h_3, h_4)$. $(sigk, vk) \xleftarrow{\$} \mathsf{SGen}(1^\kappa)$; $z \xleftarrow{\$} \mathbb{Z}_p$; $c_1 \leftarrow m \cdot \mathbf{e}(g_1, g_2)^z$; $c_2 \leftarrow g^z$; $c_\omega \leftarrow H_{g_2,\mathsf{J},h_1,h_2,h_3,h_4}(\omega)^z$; $c_t \leftarrow H_{g_2,\mathsf{J},h_1,h_2,h_3,h_4}(t)^z$; $c_{vk} \leftarrow H_{g_2,h_1,h_2,h_3,h_4}(vk)^z$; $c \leftarrow (\omega, t, c_\omega, c_t, c_{vk}, c_1, c_2)$; $\sigma \leftarrow \mathcal{T}(sigk, c)$. Return $\tilde{c} = (c, \sigma, vk)$.

- **Decryption** $\mathcal{D}'(dk_{\omega,t}, \tilde{c})$: Parse $dk_{\omega,t}$ as (D, E, d, e), $\tilde{c}$ as $(c = (\omega, t, c_\omega, c_t, c_{vk}, c_1, c_2), \sigma, vk)$.

  If $\mathsf{Ver}(vk, c, \sigma) \neq 1$, then return $\perp$.

  Else pick $r_1, r_2, r_3 \xleftarrow{\$} \mathbb{Z}_p$

  If $\mathbf{e}(c_2, H_{g_2,\mathsf{J},h_1,h_2,h_3,h_4}(\omega)^{r_1} \cdot H_{g_2,\mathsf{J},h_1,h_2,h_3,h_4}(t)^{r_2} \times H_{g_2,\mathsf{J},h_1,h_2,h_3,h_4}(vk)^{r_3}) \neq$

  $\mathbf{e}(g, c_\omega^{r_1} c_t^{r_2} c_{vk}^{r_3})$, then return $\perp$.

  Else $m \leftarrow c_1 \left(\frac{\mathbf{e}(d,c_\omega)}{\mathbf{e}(D,c_2)}\right)^{\frac{t}{t-\omega}} \left(\frac{\mathbf{e}(e,c_t)}{\mathbf{e}(E,c_2)}\right)^{\frac{\omega}{\omega-t}}$

  Return m.

One can verify that consistency follows directly from consistency of $\mathcal{OTS}$ and $\mathcal{RIBE}$.

$RIBE_{CCA}$ Security. We claim the following.

Theorem 5.3.1. Let $\mathcal{G}$ be a prime order bilinear group generator, $\mathcal{OTS} = (\mathsf{SGen}, \mathcal{T}, \mathsf{Ver})$ be a one-time signature scheme and $\mathcal{RIBE}_{CCA}[\mathcal{G}, \mathcal{OTS}] = (\mathcal{S}', \mathcal{SK}', \mathcal{KU}', \mathcal{DK}, \mathcal{E}', \mathcal{D}', \mathcal{R})$ be the associated Revocable IBE scheme as per construction above. Then for any adversary A attacking sRID-CCA security of $\mathcal{RIBE}$ with n users, whose running time is t_A and who asks q_p private key generation queries, q_k key update generation queries, q_r revocation

queries and q_d decryption queries, there exist an adversary B solving DBDH problem for $\mathcal{G}$, and an adversary F attacking strong unforgeability of $\mathcal{OTS}$ such that

$$\mathbf{Adv}^{\text{srid-cca}}_{\mathcal{RIBE},n}(A) \leq 4 \cdot \mathbf{Adv}^{\text{dbdh}}_{\mathcal{G}}(B) + \mathbf{Adv}^{\text{suf-ots}}_{\mathcal{OTS}}(F) \ , \tag{13}$$

where the last term refers to the advantage of F breaking strong unforgeability of $\mathcal{OTS}$ (cf. [32] for the definition) and $t_B \approx t_F \approx t_A + q_d(\frac{2t_p}{\log n}) + O(k^3)$.

We provide some intuition before giving the actual proof. It is not hard to show that $\mathcal{RIBE}_{CCA}$ is sRID-CPA secure, the security proof is very similar to the proof of Theorem 5.2.2. Even though a ciphertext is encrypted under an additional attribute: the verification key, the key authority never issues the corresponding decryption key component. To show that $\mathcal{RIBE}_{CCA}$ is also sRID-CCA secure, the simulator (the DBDH adversary) needs to simulate the decryption oracle. Using the arguments very similar to those used in [52], we can show that the randomized check in the decryption algorithm that the simulator can perform as well, does guarantee with overwhelming probability that a ciphertext was formed correctly (according to the encryption algorithm). If the adversary queries a ciphertext whose verification key component is the same as that of the challenge ciphertext, then the decryption query cannot be answered correctly, but in this case one can construct an adversary breaking security of the one-time signature scheme. If the verification keys are different and the ciphertext passes the randomized check, then the simulator can generate the decryption key corresponding to the identity and the verification key of the queried ciphertext, and such a decryption key will successfully decrypt the ciphertext, because we know from the randomized check that the queried ciphertext is a well-formed ciphertext. Generating such a decryption key is possible, because the verification key is different from the challenge verification key, and following the proof of security of Fuzzy IBE, it is possible for the simulator to generate valid keys for a set of attributes, if they overlap with the challenge set of attributes in fewer than the threshold number of attributes.

Proof of Theorem 5.3.1. We construct an adversary B for the DBDH problem associated with $\mathcal{G}$. B gets $(1^\kappa, \tilde{\mathbb{G}}, p, g, X, Y, Z, W)$ as input, and it has to return a bit d. It is going to use A.

$B(1^\kappa, \tilde{\mathbb{G}}, p, g, X, Y, Z, W)$:

$(\omega^*, t^*, state) \xleftarrow{\$} A(1^\kappa)$

$g_1 \leftarrow X,\ g_2 \leftarrow Y$

$sigk^*, vk^* \xleftarrow{\$} \mathsf{SGen}(1^\kappa),\ uk^* \leftarrow 01||vk^*,\ l_{uk^*} \xleftarrow{\$} \mathbb{Z}_p$

Pick random third-degree polynomials $f(x), u(x)$ with coefficients in $\mathbb{Z}_p$,

s.t. $u(x) = -x^3$ for $x = \omega^*, t^*, uk^*$, o.w. $u(x) \neq -x^3$

For $i = 1, 2, 3, 4$: $h_i \leftarrow g_2^{u(i)} g^{f(i)}$

$pk = (g, g_1, g_2, h_1, h_2, h_3, h_4)$

Let rl be an empty set and T be a binary tree with at least n leaf nodes

Pick a leaf node v^* from T and a random bit rev.

Run A and answer its queries to the $\mathcal{SK}'(\cdot), \mathcal{KU}'(\cdot), \mathcal{R}(\cdot,\cdot), \mathcal{D}'(\cdot)$ oracles using subroutines defined below

$(m_0, m_1, state') \xleftarrow{\$} A^{\mathcal{SK}'(\cdot),\mathcal{KU}'(\cdot),\mathcal{R}(\cdot,\cdot),\mathcal{D}'(\cdot)}(1^\kappa, pk, state)$

$b \xleftarrow{\$} \{0, 1\}$

$c^*_\omega \leftarrow Z^{f(\omega^*)},\ c^*_t \leftarrow Z^{f(t^*)}, c^*_{uk} \leftarrow Z^{f(uk^*)}, c^*_1 \leftarrow m_b \cdot W, c^*_2 \leftarrow Z$

$c^* \leftarrow (\omega^*, t^*, c^*_\omega, c^*_t, c^*_{uk}, c^*_1, c^*_2)$

$\sigma^* \leftarrow \mathcal{T}(sigk^*, c^*)$.

$\tilde{c}^* \leftarrow (c^*, \sigma^*, vk^*)$.

$d \xleftarrow{\$} A^{\mathcal{SK}(\cdot),\mathcal{KU}(\cdot),\mathcal{R}(\cdot,\cdot),\mathcal{D}(\cdot)}(1^\kappa, pk, \tilde{c}^*, state')$

If any of the oracles abort, return 1

Else if $b = d$, return 1, else return 0

Now we present the subroutines for answering the oracle queries. Subroutines for $\mathcal{SK}'(\cdot), \mathcal{KU}'(\cdot), \mathcal{R}(\cdot,\cdot)$ oracles are very similar to those in the Proof of Theorem 5.2.2, but we give them here too for the sake of completeness. The oracles use the functions that we

define here.

For $i, j, l, r' \in \mathbb{Z}_p,\ uk \leftarrow uk^*,\ l_{uk} \leftarrow l_{uk^*},\ S = \{0, j, uk\}$ define

$$F_1(i, j, l, r') \stackrel{\text{def}}{=} g_2^{l\Delta_{j,S}(i)+l_{uk}\Delta_{uk,S}(i)} \left(g_1^{\frac{-f(i)}{i^3+u(i)}} \left(g_2^{i^3+u(i)} g^{f(i)} \right)^{r'} \right)^{\Delta_{0,S}(i)}$$

$$F_2(i, r') \stackrel{\text{def}}{=} \left(g_1^{\frac{-1}{i^3+u(i)}} g^{r'} \right)^{\Delta_{0,S}(i)} .$$

$\mathcal{SK}'(\omega)$:

if $rev = 0$ and $\omega = \omega^*$, then abort

if $rev = 1$ and $\omega = \omega^*$, then

$v \leftarrow v^*$,

$\forall x \in \mathsf{Path}(v)\ r_x \stackrel{\$}{\leftarrow} \mathbb{Z}_p$

if $\nexists l_x$, then $l_x \stackrel{\$}{\leftarrow} \mathbb{Z}_p$, and store l_x in node x

$D_x \leftarrow g_2^{l_x} H_{g_2,h_1,h_2,h_3,h_4}(\omega^*)^{r_x},\ d_x \leftarrow g^{r_x}$

if $rev = 0$ and $\omega \neq \omega^*$, then

pick an unassigned leaf node v from T, and store ω in node v

$\forall x \in \mathsf{Path}(v)\ r'_x \stackrel{\$}{\leftarrow} \mathbb{Z}_p$

if $\nexists l_x$, then $l_x \stackrel{\$}{\leftarrow} \mathbb{Z}_p$, and store l_x in node x

$i \leftarrow \omega, j \leftarrow t^*, l \leftarrow l_x, r' \leftarrow r'_x$

$D_x \leftarrow F_1(i, j, l, r'),\ d_x \leftarrow F_2(i, r')$

if $rev = 1$ and $\omega \neq \omega^*$, then:

pick an unassigned leaf node v from T, and store ω in node v

$\forall x \in \mathsf{Path}(v)\ r'_x \stackrel{\$}{\leftarrow} \mathbb{Z}_p$

if $\nexists l_x$, then $l_x \stackrel{\$}{\leftarrow} \mathbb{Z}_p$, and store l_x in node x

$i \leftarrow \omega, l \leftarrow l_x, r' \leftarrow r'_x$

$\forall x \in (\mathsf{Path}(v) \setminus \mathsf{Path}(v^*))$

$j \leftarrow t^*$

$D_x \leftarrow F_1(i, j, l, r'),\ d_x \leftarrow F_2(i, r')$

$\forall x \in (\mathsf{Path}(v) \cap \mathsf{Path}(v^*))$

$j \leftarrow \omega^*$

$D_x \leftarrow F_1(i, j, l, r'),\ d_x \leftarrow F_2(i, r')$

return $sk_\omega = \{(x, D_x, d_x)\}_{x \in \mathsf{Path}(v)}$

$\mathcal{R}(\omega, t)$:

for all leaf nodes v associated with identity ω, add (v, t) to rl

$\mathcal{KU}'(t)$:

if $rev = 1, t = t^*$ and $\forall t \leq t^*\ (\omega^*, t) \notin rl$, then abort

else if $t = t^*$, then:

$\forall x \in \mathsf{KUNodes}(\mathsf{T}, rl, t), r_x \xleftarrow{\$} \mathbb{Z}_p$

$E_x \leftarrow g_2^{l_x} H_{g_2,h_1,h_2,h_3,h_4}(t^*)^{r_x},\ e_x \leftarrow g^{r_x}$

if $rev = 1, t \neq t^*$, then

$\forall x \in (\mathsf{KUNodes}(\mathsf{T}, rl, t) \setminus \mathsf{Path}(v^*))$

$r'_x \xleftarrow{\$} \mathbb{Z}_p, i \leftarrow t, j \leftarrow t^*, l \leftarrow l_x, r' \leftarrow r'_x$

$E_x \leftarrow F_1(i, j, l, r'),\ e_x \leftarrow F_2(i, r')$

$\forall x \in (\mathsf{KUNodes}(\mathsf{T}, rl, t) \cap \mathsf{Path}(v^*))$

$r'_x \xleftarrow{\$} \mathbb{Z}_p, i \leftarrow t, j \leftarrow \omega^*, l \leftarrow l_x, r' \leftarrow r'_x$

$E_x \leftarrow F_1(i, j, l, r'),\ e_x \leftarrow F_2(i, r')$

if $rev = 0, t \neq t^*$, then

$\forall x \in \mathsf{KUNodes}(\mathsf{T}, rl, t)$

$r'_x \xleftarrow{\$} \mathbb{Z}_p, i \leftarrow t, j \leftarrow t^*, l \leftarrow l_x, r' \leftarrow r'_x$

$E_x \leftarrow F_1(i, j, l, r'),\ e_x \leftarrow F_2(i, r')$

return $ku_t = \{(x, E_x, e_x)\}_{x \in \mathsf{KUNodes}(\mathsf{T}, rl, t)}$

$\mathcal{D}'(\tilde{c})$:

parse $\tilde{c}$ as $(c = (\omega, t, c_\omega, c_t, c_{uk}, c_1, c_2), \sigma, vk)$

we assume that $\omega = \omega^*, t = t^*$, o.w. such a decryption query can be trivially replied by generating a decryption key using $\mathcal{SK}(\omega)$ and $\mathcal{KU}(t)$ oracles

if $\mathsf{Ver}(vk, c, \sigma) \neq 1$ (i.e. invalid ciphertext), then abort

if $\mathsf{Ver}(vk, c, \sigma) = 1$ and $vk = vk^*$, then return a random bit

if $\mathsf{Ver}(vk, c, \sigma) = 1$ and $vk \neq vk^*$, then

first do a ciphertext sanity check on c (as explained in Section 5.3)

if c passes sanity check, then generate $dk_{\omega,uk} = (D, E, d, e)$ as follows

$uk \leftarrow 01 \| vk,\ r_\omega, l_\omega, l_t, r'_{uk} \xleftarrow{\$} \mathbb{Z}_p,\ S \leftarrow \{0, \omega, t\}$

$D \leftarrow g_2^{l_\omega} H_{g_2,h_1,h_2,h_3,h_4}(\omega)^{r_\omega},\ d \leftarrow g^{r_\omega}$

$E \leftarrow \left(g_2^{l_\omega \Delta_{\omega,S}(uk) + l_t \Delta_{t,S}(uk)}\right) \cdot \left(g_1^{\frac{-f(uk)}{uk^3+u(uk)}} \left(g_2^{uk^3+u(uk)} g^{f(uk)}\right)^{r'_{uk}}\right)^{\Delta_{0,S}(uk)}$

$e \leftarrow \left(g_1^{\frac{-1}{uk^3+u(uk)}} g^{r'_{uk}}\right)^{\Delta_{0,S}(uk)}$

now use $dk_{\omega,uk}$ to decrypt c as follows

$m \leftarrow c_1 \left(\frac{\mathbf{e}(d,c_\omega)}{\mathbf{e}(D,c_2)}\right)^{\frac{uk}{uk-\omega}} \left(\frac{\mathbf{e}(e,c_{uk})}{\mathbf{e}(E,c_2)}\right)^{\frac{\omega}{\omega-uk}}$

return m

Analysis. Justification mostly follows directly from the Proof of Theorem 5.2.2, except for the justification for the simulation of the decryption oracle that we present here. Note that in $\mathcal{RIBE}_{CCA}$, only 2 out of 3 attributes need to be matched between a ciphertext and decryption key for successful decryption, as opposed to 2 out of 2 attributes in $\mathcal{RIBE}$. Therefore, in the decryption algorithm, the ciphertext component corresponding to the verification-key attribute is used only for the ciphertext sanity check and not for actual decryption. However, the decryption oracle needs to use this ciphertext component for decrypting ciphertexts encrypted for the challenge identity and the challenge time. The decryption oracle will therefore fail to decrypt malformed ciphertexts, in particular those ciphertexts whose verification key components are invalid. So, the ciphertext sanity check

which discards all such ciphertexts with a very high probability, ensures that the decryption oracle will be able to answer all but a negligible fraction of valid decryption queries. Let $q(y)$ be a second-degree polynomial. Let $g_1 = g^a$ (to simplify notation). Define $q(0) = a,\ q(\omega) = l_\omega, q(t) = l_t,\ r_{uk} = (r'_{uk} - \frac{a}{uk^3+u(uk)})\Delta_{0,S}(uk)$, then as in Section 6.3 of [83], we can show that $D = g_2^{q(\omega)} H_{g_2,h_1,h_2,h_3,h_4}\omega)^{r_\omega},\ d = g^{r_\omega},\ E = g_2^{q(uk)} H_{g_2,h_1,h_2,h_3,h_4}(uk)^{r_{uk}},\ e = g^{r_{uk}}$. From the consistency condition of $\mathcal{RIBE}_{CCA}$, it follows that decryption of c using $dk_{\omega,uk}$ will yield the right message. Hence, B correctly simulates the decryption oracle, except for the event when A queries a ciphertext (c, σ, vk^*) to the decryption oracle, where $(c, \sigma) \neq (c^*, \sigma^*)$ and $\mathsf{Ver}(vk^*, c, \sigma) = 1$. We denote this event by forge. Note that the definition of forge is essentially the same as in the Proof of Theorem 1 of [32], but we write it here too for the sake of completeness. Event forge also includes the case where A queries (c, σ, vk^*) before receiving the challenge ciphetext, in which case (c, σ) may or may not be equal to (c^*, σ^*). Now, it is easy to see that (c, σ, vk^*) can be used to forge a signature of $\mathcal{OTS}$ with probability $\Pr[\,\mathsf{forge}\,]$. Namely, one can construct an adversary F, such that $\mathbf{Adv}^{\text{suf-ots}}_{\mathcal{OTS}}(F) \geq \Pr[\,\mathsf{forge}\,]$.

Thus, from Theorem 5.2.2 and decryption oracle oracle simulation we have

$$\begin{aligned}\mathbf{Adv}^{\text{srid-cca}}_{\mathcal{RIBE}_{CCA},n}(A) &\leq 4 \cdot \mathbf{Adv}^{\text{dbdh}}_{\mathcal{G}}(B) + \Pr[\,\mathsf{forge}\,] \\ &\leq 4 \cdot \mathbf{Adv}^{\text{dbdh}}_{\mathcal{G}}(B) + \mathbf{Adv}^{\text{suf-ots}}_{\mathcal{OTS}}(F)\ .\end{aligned}$$

The above implies the statement of the theorem. □

We note that alternatively we could utilize simulation-sound NIZK proofs in a way similar to the construction of CCA secure Fuzzy IBE in [83], but our construction is more efficient.

Generic CCA construction. The Fujisaki-Okamoto (or, FO for short) transform [44, 43] is a generic transform to convert a CPA secure public-key encryption scheme into a CCA secure one, in the RO model. The transform can also be applied to IBE schemes as shown in [64]. Here we show how to apply the FO transform to Revocable IBE schemes. Unlike the

previous approach, this solution is generic in that it applies to any Revocable IBE scheme. If applied to our construction, then we suggest to use its more efficient RO modification we discussed, since the FO transform also relies on the RO model.

Let $\mathcal{RIBE} = (\mathcal{S}, \mathcal{SK}, \mathcal{KU}, \mathcal{DK}, \mathcal{E}, \mathcal{D}, \mathcal{R})$ be any Revocable IBE scheme as per Definition 5.1.1. We can construct another Revocable IBE scheme $\mathcal{FO}\text{-}\mathcal{RIBE}_{CCA} = (\mathcal{S}', \mathcal{SK}, \mathcal{KU}, \mathcal{DK}, \mathcal{E}', \mathcal{D}', \mathcal{R})$ as follows (we only specify the differences from $\mathcal{RIBE}$). Let $(\mathsf{MsgSp}, \mathsf{IdSp}, \mathsf{TimeSp})$ and $(\mathsf{MsgSp}', \mathsf{IdSp}, \mathsf{TimeSp})$ be the (message space, identity space, time space) of $\mathcal{RIBE}$ and $\mathcal{FO}\text{-}\mathcal{RIBE}_{CCA}$, respectively. Let $\mathcal{COINS}$ be the set from where $\mathcal{E}$ draws its random coins. We require that for every $m \in \mathsf{MsgSp}', \sigma \in \{0,1\}^\kappa$, we have that $m \,||\, \sigma \in \mathsf{MsgSp}$. To make the use of randomness explicit, we use the notation $r \xleftarrow{\$} \mathcal{COINS}; \mathcal{E}(\cdot,\cdot,\cdot,\cdot;r)$ as opposed to the traditional shorthand $\mathcal{E}(\cdot,\cdot,\cdot,\cdot)$. The setup algorithm $\mathcal{S}'(1^\kappa, n)$ follows $\mathcal{S}$. In addition, it specifies a hash function $\mathcal{H} : \{0,1\}^* \rightarrow \mathcal{COINS}$ and outputs it as part of the public parameters pk'. The encryption and decryption algorithms are as follows.

- **Encryption** $\mathcal{E}'(pk', \omega, t, m)$: $\sigma \xleftarrow{\$} \{0,1\}^\kappa$; $r \leftarrow \mathcal{H}(m||\sigma||\omega||t)$; $c \leftarrow \mathcal{E}(pk, \omega, t, m||\sigma; r)$. Return c.

- **Decryption** $\mathcal{D}'(dk_{\omega,t}, c)$: $m' \leftarrow \mathcal{D}(dk_{\omega,t}, c)$. Parse m' as $m||\sigma$; $r \leftarrow \mathcal{H}(m||\sigma \,||\, \omega \,||\, t)$. If $c = \mathcal{E}(pk, \omega, t, m||\sigma\ ;\ r)$, then return m, else return $\perp$.

Consistency follows from the justification of the consistency requirement for $\mathcal{RIBE}$.

Before we present the security analysis of $\mathcal{FO}\text{-}\mathcal{RIBE}_{CCA}$, we define a property that we call γ-uniformity for Revocable IBE schemes.

γ-UNIFORMITY. The γ-uniformity property has been defined earlier in the context of public key encryption schemes [44] and identity-based encryption schemes [90]. Here we define it for Revocable IBE schemes.

Definition 5.3.2. [γ-uniformity] Let $\mathcal{RIBE} = (\mathcal{S}, \mathcal{SK}, \mathcal{KU}, \mathcal{DK}, \mathcal{E}, \mathcal{D}, \mathcal{R})$ be a Revocable IBE scheme. For any public parameter pk output by $\mathcal{S}$, any given identity $\omega \in \mathsf{IdSp}$, time

$t \in \mathsf{TimeSp}$, the corresponding decryption key dk, message $m \in \mathsf{MsgSp}$ and a ciphertext c output by $\mathcal{E}$ define

$$\gamma(m, c) = \Pr\left[r \xleftarrow{\$} \mathcal{COINS} : c = \mathcal{E}(pk, \omega, t, m; r) \right].$$

We say that $\mathcal{RIBE}$ is γ-uniform if for any $\omega \in \mathsf{IdSp}, t \in \mathsf{TimeSp}, m \in \mathsf{MsgSp}$ and any ciphertext c output by $\mathcal{E}$, $\gamma(m, c) \leq \gamma$.

$\mathcal{FO}$-$\mathcal{RIBE}_{CCA}$ Security.

Theorem 5.3.3. Let $\mathcal{RIBE}$ be a γ-uniform Revocable IBE scheme with message space MsgSp, identity space IdSp, time space TimeSp and set of coins $\mathcal{COINS}$ for its encryption algorithm. Let $\mathcal{H}$ be a hash function mapping $\mathsf{MsgSp} \times \mathsf{IdSp} \times \mathsf{TimeSp}$ to $\mathcal{COINS}$ (modeled as the RO) and $\mathcal{FO}$-$\mathcal{RIBE}_{CCA}$ be the associated Revocable IBE scheme as per construction above. Then for an adversary A attacking sRID-CCA security of $\mathcal{FO}$-$\mathcal{RIBE}_{CCA}$ with n users, whose running time is t_A and who asks q_d decryption queries and q_h random oracle queries, there exists an adversary B attacking sRID-CPA security of $\mathcal{RIBE}$ such that

$$\mathbf{Adv}^{\text{srid-cca}}_{\mathcal{FO}\text{-}\mathcal{RIBE}_{CCA},n}(A) \leq \left(\mathbf{Adv}^{\text{srid-cpa}}_{\mathcal{RIBE},n}(B) + \frac{1}{2} \right) \cdot \left(\frac{1}{1 - \gamma q_d} \right) + \frac{q_h}{2^\kappa} - \frac{1}{2}\,, \tag{14}$$

where the running time of B, $t_B \approx t_A + \kappa q_h$.

Proof of Theorem 5.3.3. We construct an adversary B attacking the sRID-CPA security of $\mathcal{RIBE}$. B is going to use the adversary A attacking sRID-CCA security of $\mathcal{FO}$-$\mathcal{RIBE}_{CCA}$ (in RO model). It has access to $\mathcal{SK}(\cdot), \mathcal{KU}(\cdot)$, and $\mathcal{R}(\cdot, \cdot)$ oracles. It needs to simulate random oracle $\mathcal{RO}(\cdot)$ (for hash function $\mathcal{H}$) and decryption oracle $\mathcal{D}'(\cdot)$.

$B^{\mathcal{SK}(\cdot),\mathcal{KU}(\cdot),\mathcal{R}(\cdot,\cdot)}(1^\kappa)$:

$(\omega^*, t^*, state) \xleftarrow{\$} A(1^\kappa)$

return $(\omega^*, t^*, state)$ and get back pk

run A and answer its queries to the $\mathcal{SK}(\cdot), \mathcal{KU}(\cdot)$ and $\mathcal{R}(\cdot, \cdot)$ oracles by using own oracles. For $\mathcal{RO}(\cdot)$ and $\mathcal{D}'(\cdot)$ oracles use subroutines defined below

$(m_0, m_1, state') \xleftarrow{\$} A^{\mathcal{SK}(\cdot),\mathcal{KU}(\cdot),\mathcal{R}(\cdot,\cdot),\mathcal{D}'(\cdot),\mathcal{RO}(\cdot)}(1^\kappa, pk, state)$

$\sigma_0, \sigma_1 \xleftarrow{\$} \{0,1\}^\kappa$

return $(m_0||\sigma_0, m_1||\sigma_1, state')$ and get back c^*

$d \xleftarrow{\$} A^{\mathcal{SK}(\cdot),\mathcal{KU}(\cdot),\mathcal{R}(\cdot,\cdot),\mathcal{D}'(\cdot),\mathcal{RO}(\cdot)}(1^\kappa, pk, c^*, state')$

if any of the oracles or A aborts, then for $i \in \{0,1\}$ find a tuple $(m_i||\sigma_i, \omega, t, r)$

in the list stored by $\mathcal{RO}$. If such a tuple is found, return i, else return 1

else if neither the oracles nor A aborts, return d

Below we present the subroutines for answering $\mathcal{RO}(\cdot)$ and $\mathcal{D}'(\cdot)$ oracle queries.

$\mathcal{RO}(m \,||\, \omega \,||\, t)$:

if (m, ω, t) was queried before, then return the corresponding r

else $r \xleftarrow{\$} \mathcal{COINS}$, store (m, ω, t, r), and return r

$\mathcal{D}'(c)$:

compute ω, t from c

find a tuple (m, ω, t, r) stored by $\mathcal{RO}(\cdot)$ oracle, such that $c = \mathcal{E}(pk, \omega, t, m\ ;\ r)$

if no such pair found, then abort

else parse m as $m'||\sigma$, where $m' \in \mathsf{MsgSp}'$ and $\sigma \in \{0,1\}^\kappa$

return m'

Analysis. Let the challenge bit for the adversary B be b. Define the following events.

abort: Decryption oracle aborts in the sRID-CCA experiment simulated by B.

succA: A succeeds in the sRID-CCA experiment simulated by B, given that abort does not occur.

succB: B succeeds in the sRID-CPA experiment, given that abort does not occur.

queryb: A queries $(m_b||\sigma_b, \omega, t, r)$ to the random oracle in the sRID-CCA experiment simulated by B.

queryb′: A queries $(m_{\bar{b}}||\sigma_{\bar{b}}, \omega, t, r)$ to the random oracle in the sRID-CCA experiment simulated by B.

$$\begin{aligned}
\Pr[\,\mathsf{succA}\,] \;=\; & \Pr[\,\mathsf{succA} \mid \mathsf{queryb}\,]\Pr[\,\mathsf{queryb}\,] \\
& +\Pr[\,\mathsf{succA} \mid \neg\mathsf{queryb} \wedge \mathsf{queryb}'\,]\Pr[\,\neg\mathsf{queryb} \wedge \mathsf{queryb}'\,] \\
& +\Pr[\,\mathsf{succA} \mid \neg\mathsf{queryb} \wedge \neg\mathsf{queryb}'\,]\Pr[\,\neg\mathsf{queryb} \wedge \neg\mathsf{queryb}'\,]
\end{aligned}$$

$$\begin{aligned}
\Pr[\,\mathsf{succB}\,] \;=\; & \Pr[\,\mathsf{succB} \mid \mathsf{queryb}\,]\Pr[\,\mathsf{queryb}\,] \\
& +\Pr[\,\mathsf{succB} \mid \neg\mathsf{queryb} \wedge \mathsf{queryb}'\,]\Pr[\,\neg\mathsf{queryb} \wedge \mathsf{queryb}'\,] \\
& +\Pr[\,\mathsf{succB} \mid \neg\mathsf{queryb} \wedge \neg\mathsf{queryb}'\,]\Pr[\,\neg\mathsf{queryb} \wedge \neg\mathsf{queryb}'\,] \;.
\end{aligned}$$

From the description of B, we have

$$\begin{aligned}
&\Pr[\,\mathsf{succB} \mid \mathsf{queryb}\,] = 1 \;, \\
&\Pr[\,\mathsf{succB} \mid \mathsf{queryb}'\,] = 0 \;, \\
&\Pr[\,\mathsf{succA} \mid \neg\mathsf{queryb} \wedge \neg\mathsf{queryb}'\,] = \Pr[\,\mathsf{succB} \mid \neg\mathsf{queryb} \wedge \neg\mathsf{queryb}'\,] \;.
\end{aligned}$$

Thus, we have

$$\begin{aligned}
\Pr[\,\mathsf{succB}\,] - \Pr[\,\mathsf{succA}\,] \;=\; & (1 - \Pr[\,\mathsf{succA} \mid \mathsf{queryb}\,])\Pr[\,\mathsf{queryb}\,] \\
& -\Pr[\,\mathsf{succA} \mid \neg\mathsf{queryb} \wedge \mathsf{queryb}'\,]\Pr[\,\neg\mathsf{queryb} \wedge \mathsf{queryb}'\,] \\
\geq\; & -\Pr[\,\neg\mathsf{queryb} \wedge \mathsf{queryb}'\,] \;.
\end{aligned}$$

But we know that the probability that A queries a $(m_{\bar{b}}||\sigma_{\bar{b}}, \omega, t, r)$ tuple to the random oracle is at most $2^{-\kappa}$, because $\sigma_{\bar{b}}$ is a randomly chosen κ-bit value that is information theoretically hidden from A. So,

$$\Pr[\,\neg\mathsf{queryb} \wedge \mathsf{queryb}'\,] \leq \frac{q_h}{2^{\kappa}}.$$

Also, $\Pr[\,\mathsf{succA}\,] = \mathbf{Adv}^{\text{srid-cca}}_{\mathcal{FO}\text{-}\mathcal{RIBE}_{CCA},n}(A) + 1/2$. So,

$$\Pr[\,\mathsf{succB}\,] \geq \mathbf{Adv}^{\text{srid-cca}}_{\mathcal{FO}\text{-}\mathcal{RIBE}_{CCA},n}(A) + \frac{1}{2} - \frac{q_h}{2^{\kappa}} \;.$$

It remains to estimate $\Pr[\neg\mathsf{abort}]$. The event abort occurs when A queries the decryption oracle for some ciphertext without querying the random oracle for the randomness that was used to generate the ciphertext. From the definition of γ-uniformity (Definition 5.3.2), we know that this can happen with probability at most γ. Thus,

$$\Pr[\neg\mathsf{abort}] \leq (1-\gamma)^{q_d} \approx (1-\gamma q_d) .$$

Therefore,

$$\mathbf{Adv}^{\text{srid-cpa}}_{\mathcal{RIBE},n}(B) \geq (1-\gamma q_d)\left(\mathbf{Adv}^{\text{srid-cca}}_{\mathcal{FO}\text{-}\mathcal{RIBE}_{CCA},n}(A) + \frac{1}{2} - \frac{q_{ro}}{2^{\kappa}}\right) - \frac{1}{2} .$$

Thus, Equation (14) follows from the above,

$$\mathbf{Adv}^{\text{srid-cca}}_{\mathcal{FO}\text{-}\mathcal{RIBE}_{CCA},n}(A) \leq \left(\mathbf{Adv}^{\text{srid-cpa}}_{\mathcal{RIBE},n}(B) + \frac{1}{2}\right)\left(\frac{1}{1-\gamma q_d}\right) + \frac{q_{ro}}{2^{\kappa}} - \frac{1}{2} .$$

The above implies the statement of the theorem. □

5.4 Revocable ABE and Fuzzy IBE

Key-policy attribute-based encryption (KP-ABE) [53] is a generalization of Fuzzy IBE, which allows the key authority to specify more advanced decryption policies. In KP-ABE, as in Fuzzy IBE, each ciphertext is labeled by the sender with a set of descriptive attributes. However, each private key is associated with an access tree that specifies which type of ciphertexts the key can decrypt. A key can decrypt a particular ciphertext, only if the ciphertext attributes satisfy the access tree of the key. The problem of revocation of attributes is as relevant to KP-ABE as the problem of identity revocation is relevant for IBE. There is no solution known other than the frequent key update for all attributes. As we explained earlier, this solution does not scale well. We extend our ideas to construct a *key-policy attribute-based encryption with efficient revocation*, or simply, *Revocable KP-ABE*. We will only explain how to obtain a Revocable KP-ABE, and that will imply a Revocable Fuzzy IBE as well. The security of our construction holds only in a weaker selective revocation list model, where the adversary must declare in advance all the users to be revoked prior to the challenge time.

The construction uses the KP-ABE construction from [53] and a binary tree in the following way. Messages are encrypted with attributes β and time, where β is the set of attributes which is used in encryption in KP-ABE. The root node of the access tree of decryption key is a 2-out-of-2 gate whose one child is time (similarly to Revocable IBE) and the other child is the root node of the access tree. The component of decryption key corresponding to the access tree and the time are called private key and key update, respectively. Private key for an access tree is computed in the same way as keys are computed in KP-ABE, except that instead of the root polynomial of the access tree, the root polynomial of decryption key evaluates to the master key at 0. The use of binary tree is essentially the same in both Revocable IBE and Revocable KP-ABE, e.g., the way users are assigned to leaf nodes, the way polynomials are selected for each node, the number of private keys each user gets, the way key updates are computed, etc.

CHAPTER VI

CONCLUSIONS

Even though this thesis was mainly motivated by provable security in practice, only the first result has a direct impact on a practical and widely deployed protocol Kerberos, in that this is the first work on provable security analysis of the two types of authenticated encryption schemes used in Kerberos version 5, called General Profile and Simplified Profile. Our results show that the authenticated encryption paradigm used in General Profile does not provide integrity, even if it uses secure building blocks (e.g., a collision-resistant hash function and an IND-CPA encryption scheme). While our attacks do not apply for particular instantiations of General Profile suggested in the specifications, they do show limitation of the design. We suggest simple and easy to implement modifications, and we show that the resulting scheme provably provides privacy and authenticity, under standard assumptions. We prove that Modified General Profile and Simplified Profile are IND-CCA and INT-CTXT secure, if they utilize secure building blocks. This justifies the assumption about the security of encryption necessary for the recent formal-methods-based symbolic analyses. Together, these results provide strong security guarantees for Kerberos that we believe will help its standardization.

Our second result concerns the generation of different strings, e.g., confounder, session keys, sequence numbers, etc., which are all suggested to be generated at random in the Kerberos specifications. Our proposed hash-function-based PRG can be used to generate these strings with provable security guarantees. The security of our PRG relies on the assumptions that the underlying hash function is regular and collision-resistant, where the regularity assumption can be relaxed to a new notion of worst-case regularity introduced in our work, and the collision-resistance is required to be exponential. We note that even

though our PRG is the most efficient function-based PRG known that relies on reasonable assumptions, it is still relatively less efficient than practical and standardized PRGs, and as such we don't envision it being directly used in practical protocols. We believe that the small loss in efficiency is justified, given that our assumptions are much more standard and reasonable compared to the ones used for standardized PRGs.

Finally, in our third and last result, we present new primitives, and their efficient and provably secure constructions targeted at the problem of revocation in IBEs and ABEs. Due to the attractive features offered by these new types of encryption schemes, they have been used in many practical protocols, and are most likely to be used in future as well. We note that our results and techniques for revocation have found much wider applications in information security, such as mobile social networks, cloud-based secure health records, data outsourcing systems, vehicular ad-hoc networks, etc.

REFERENCES

[1] Abadi, M. and Rogaway, P., "Reconciling Two Views of Cryptography (The Computational Soundness of Formal Encryption)," *J. Cryptology*, vol. 20, no. 3, p. 395, 2007.

[2] Aiello, W., Lodha, S., and Ostrovsky, R., "Fast Digital Identity Revocation (Extended Abstract)," in *CRYPTO*, pp. 137–152, 1998.

[3] ANSI, "ANSI X9.17 (Revised): American National Standard for Financial Institution Key Management (Wholesale)," 1985.

[4] Backes, M., Cervesato, I., Jaggard, A. D., Scedrov, A., and Tsay, J.-K., "Cryptographically Sound Security Proofs for Basic and Public-Key Kerberos," in *ESORICS*, pp. 362–383, 2006.

[5] Backes, M. and Pfitzmann, B., "Symmetric Encryption in a Simulatable Dolev-Yao Style Cryptographic Library," in *CSFW*, pp. 204–218, 2004.

[6] Backes, M., Pfitzmann, B., and Waidner, M., "A composable cryptographic library with nested operations," in *ACM Conference on Computer and Communications Security*, pp. 220–230, 2003.

[7] Backes, M., Pfitzmann, B., and Waidner, M., "Symmetric Authentication within a Simulatable Cryptographic Library," in *ESORICS*, pp. 271–290, 2003.

[8] Bella, G. and Paulson, L. C., "Kerberos Version 4: Inductive Analysis of the Secrecy Goals," in *ESORICS*, pp. 361–375, 1998.

[9] Bella, G. and Riccobene, E., "Formal Analysis of the Kerberos Authentication System," *J. UCS*, vol. 3, no. 12, pp. 1337–1381, 1997.

[10] Bellare, M., "New Proofs for NMAC and HMAC: Security Without Collision-Resistance," in *CRYPTO*, pp. 602–619, 2006.

[11] Bellare, M., Canetti, R., and Krawczyk, H., "Keying Hash Functions for Message Authentication," in *CRYPTO*, pp. 1–15, 1996.

[12] Bellare, M., Desai, A., Jokipii, E., and Rogaway, P., "A Concrete Security Treatment of Symmetric Encryption," in *FOCS*, pp. 394–403, 1997.

[13] Bellare, M., Kilian, J., and Rogaway, P., "The security of the cipher block chaining message authentication code," in *CRYPTO '04*, Springer, 2004.

[14] Bellare, M. and Kohno, T., "Hash Function Balance and Its Impact on Birthday Attacks," in *EUROCRYPT*, pp. 401–418, 2004.

[15] Bellare, M., Kohno, T., and Namprempre, C., "Breaking and provably repairing the SSH authenticated encryption scheme: A case study of the Encode-then-Encrypt-and-MAC paradigm," *ACM Trans. Inf. Syst. Secur.*, vol. 7, no. 2, pp. 206–241, 2004.

[16] Bellare, M. and Namprempre, C., "Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm," in *ASIACRYPT*, pp. 531–545, 2000.

[17] Bellare, M. and Namprempre, C., "Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm," *J. Cryptology*, vol. 21, no. 4, pp. 469–491, 2008.

[18] Bellare, M. and Rogaway, P., "Random Oracles are Practical: A Paradigm for Designing Efficient Protocols," in *ACM Conference on Computer and Communications Security*, pp. 62–73, 1993.

[19] Bellare, M. and Rogaway, P., "Encode-then-encipher encryption: How to exploit nonces or redundancy in plaintexts for efficient cryptography," in *ASIACRYPT '00*, Springer, 2000.

[20] Bellare, M. and Rogaway, P., "An Introduction to Modern Cryptography." UCSD CSE 207 Course Notes, 2005. Available at `http://www.cse.ucsd.edu/~mihir/cse207/index.html`.

[21] Bellare, M. and Rogaway, P., "The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs," in *EUROCRYPT*, pp. 409–426, 2006.

[22] Bellovin, S. M. and Merritt, M., "Limitations of the Kerberos Authentication System," in *USENIX Winter*, pp. 253–268, 1991.

[23] Bittau, A., Handley, M., and Lackey, J., "The Final Nail in WEP's Coffin," in *IEEE Symposium on Security and Privacy*, pp. 386–400, 2006.

[24] Blum, M. and Micali, S., "How to Generate Cryptographically Strong Sequences of Pseudo Random Bits," in *FOCS*, pp. 112–117, 1982.

[25] Boldyreva, A., Goyal, V., and Kumar, V., "Identity-based Encryption with Efficient Revocation," in *ACM Conference on Computer and Communications Security*, pp. 417–426, 2008.

[26] Boldyreva, A., Goyal, V., and Kumar, V., "Identity-based Encryption with Efficient Revocation," *IACR Cryptology ePrint Archive*, vol. 2012, 2012. Available at `http://eprint.iacr.org/2012/052`.

[27] Boldyreva, A. and Kumar, V., "Extended Abstract: Provable-Security Analysis of Authenticated Encryption in Kerberos," in *IEEE Symposium on Security and Privacy*, pp. 92–100, 2007.

[28] Boldyreva, A. and Kumar, V., "Provable-Security Analysis of Authenticated Encryption in Kerberos," *IET Inf. Secur.*, vol. 5, no. 4, pp. 207–219, 2011.

[29] Boldyreva, A. and Kumar, V., "A New Pseudorandom Generator from Collision-Resistant Hash Functions," in *CT-RSA*, pp. 187–202, 2012.

[30] Boldyreva, A. and Kumar, V., "A new pseudorandom generator from collision-resistant hash functions," *IACR Cryptology ePrint Archive*, vol. 2012, 2012. Available at `http://eprint.iacr.org/2012/056`.

[31] Boneh, D. and Boyen, X., "Efficient Selective-ID Secure Identity-Based Encryption Without Random Oracles," in *EUROCRYPT*, pp. 223–238, 2004.

[32] Boneh, D., Canetti, R., Halevi, S., and Katz, J., "Chosen-Ciphertext Security from Identity-Based Encryption," *SIAM J. Comput.*, vol. 36, no. 5, pp. 1301–1328, 2006.

[33] Boneh, D., Ding, X., Tsudik, G., and Wong, C. M., "A method for fast revocation of public key certificates and security capabilities," in *USENIX Security Symposium*, pp. 22–22, 2001.

[34] Boneh, D. and Franklin, M. K., "Identity-Based Encryption from the Weil Pairing," in *CRYPTO*, pp. 213–229, 2001.

[35] Butler, F., Cervesato, I., Jaggard, A. D., and Scedrov, A., "A Formal Analysis of Some Properties of Kerberos 5 Using MSR," in *CSFW*, pp. 175–, 2002.

[36] Butler, F., Cervesato, I., Jaggard, A. D., Scedrov, A., and Walstad, C., "Formal analysis of Kerberos 5," *Theor. Comput. Sci.*, vol. 367, no. 1-2, pp. 57–87, 2006.

[37] Canetti, R., Halevi, S., and Katz, J., "Chosen-Ciphertext Security from Identity-Based Encryption," in *EUROCRYPT*, pp. 207–222, 2004.

[38] Degabriele, J. P. and Paterson, K. G., "Attacking the IPsec Standards in Encryption-only Configurations," in *IEEE Symposium on Security and Privacy*, pp. 335–349, 2007.

[39] Degabriele, J. P. and Paterson, K. G., "On the (in)security of IPsec in MAC-then-encrypt configurations," in *ACM Conference on Computer and Communications Security*, pp. 493–504, 2010.

[40] Desai, A., Hevia, A., and Yin, Y. L., "A Practice-Oriented Treatment of Pseudorandom Number Generators," in *EUROCRYPT*, pp. 368–383, 2002.

[41] Dolev, D. and Yao, A. C.-C., "On the security of public key protocols," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–207, 1983.

[42] FIPS, "FIPS PUB 186-2, Digital Signature Standard," 1994.

[43] Fujisaki, E. and Okamoto, T., "How to Enhance the Security of Public-Key Encryption at Minimum Cost," in *Public Key Cryptography*, pp. 53–68, 1999.

[44] Fujisaki, E. and Okamoto, T., "Secure Integration of Asymmetric and Symmetric Encryption Schemes," in *CRYPTO*, pp. 537–554, 1999.

[45] Gentry, C., "Certificate-Based Encryption and the Certificate Revocation Problem," in *EUROCRYPT*, pp. 272–293, 2003.

[46] Goldreich, O., *Foundations of Cryptography - Volume 1*. Cambridge University Press, 2001.

[47] Goldreich, O., Goldwasser, S., and Micali, S., "How to construct random functions," *J. ACM*, vol. 33, no. 4, pp. 792–807, 1986.

[48] Goldreich, O., Krawczyk, H., and Luby, M., "On the Existence of Pseudorandom Generators," in *CRYPTO*, pp. 146–162, 1988.

[49] Goldreich, O. and Levin, L. A., "A Hard-Core Predicate for all One-Way Functions," in *STOC*, pp. 25–32, 1989.

[50] Goldwasser, S. and Micali, S., "Probabilistic Encryption and How to Play Mental Poker Keeping Secret All Partial Information," in *STOC*, pp. 365–377, 1982.

[51] Goyal, V., "Certificate Revocation Using Fine Grained Certificate Space Partitioning," in *Financial Cryptography*, pp. 247–259, 2007.

[52] Goyal, V., "Reducing Trust in the PKG in Identity Based Cryptosystems," in *CRYPTO*, pp. 430–447, 2007.

[53] Goyal, V., Pandey, O., Sahai, A., and Waters, B., "Attribute-based encryption for fine-grained access control of encrypted data," in *ACM Conference on Computer and Communications Security*, pp. 89–98, 2006.

[54] Haitner, I., Harnik, D., and Reingold, O., "Efficient Pseudorandom Generators from Exponentially Hard One-Way Functions," in *ICALP (2)*, pp. 228–239, 2006.

[55] Haitner, I., Harnik, D., and Reingold, O., "On the Power of the Randomized Iterate," in *CRYPTO*, pp. 22–40, 2006.

[56] Haitner, I., Reingold, O., and Vadhan, S. P., "Efficiency improvements in constructing pseudorandom generators from one-way functions," in *STOC*, pp. 437–446, 2010.

[57] Hanaoka, Y., Hanaoka, G., Shikata, J., and Imai, H., "Identity-Based Hierarchical Strongly Key-Insulated Encryption and Its Application," in *ASIACRYPT*, pp. 495–514, 2005.

[58] Håstad, J., "Pseudo-Random Generators under Uniform Assumptions," in *STOC*, pp. 395–404, 1990.

[59] Håstad, J., Impagliazzo, R., Levin, L. A., and Luby, M., "A Pseudorandom Generator from any One-way Function," *SIAM J. Comput.*, vol. 28, no. 4, pp. 1364–1396, 1999.

[60] Holenstein, T., "Pseudorandom Generators from One-Way Functions: A Simple Construction for Any Hardness," in *TCC*, pp. 443–461, 2006.

[61] Impagliazzo, R., Levin, L. A., and Luby, M., "Pseudo-random Generation from one-way functions (Extended Abstracts)," in *STOC*, pp. 12–24, 1989.

[62] Impagliazzo, R., Nisan, N., and Wigderson, A., "Pseudorandomness for network algorithms," in *STOC*, pp. 356–364, 1994.

[63] Jahid, S., Mittal, P., and Borisov, N., "EASiER: encryption-based access control in social networks with efficient revocation," in *ASIACCS*, pp. 411–415, 2011.

[64] Kitagawa, T., Yang, P., Hanaoka, G., Zhang, R., Watanabe, H., Matsuura, K., and Imai, H., "Generic Transforms to Acquire CCA-Security for Identity Based Encryption: The Cases of FOpkc and REACT," in *ACISP*, pp. 348–359, 2006.

[65] Kohno, T., "Authenticated Encryption in Practice: Generalized Composition Methods and the Secure Shell, CWC, and WinZip Schemes." UCSD Dissertation, 2006.

[66] Krawczyk, H., "The Order of Encryption and Authentication for Protecting Communications (or: How Secure Is SSL?)," in *CRYPTO '01*, pp. 310–331, Springer, 2001.

[67] Levin, L. A., "One-way functions and pseudorandom generators," *Combinatorica*, vol. 7, no. 4, pp. 357–363, 1987.

[68] Li, M., Yu, S., Ren, K., and Lou, W., "Securing Personal Health Records in Cloud Computing: Patient-Centric and Fine-Grained Data Access Control in Multi-owner Settings," in *SecureComm*, pp. 89–106, 2010.

[69] Liang, X., Li, X., Lu, R., Lin, X., and Shen, X., "An Efficient and Secure User Revocation Scheme in Mobile Social Networks," in *GLOBECOM*, pp. 1–5, 2011.

[70] Libert, B. and Quisquater, J.-J., "Efficient revocation and threshold pairing based cryptosystems," in *PODC*, pp. 163–171, 2003.

[71] Micali, S., "Efficient Certificate Revocation," *Technical Report MIT/LCS/TM-542b*, 1996.

[72] Micali, S., "Novomodo: Scalable Certificate Validation and Simplified PKI Management," in *PKI Research Workshop*, 2002.

[73] Micciancio, D. and Warinschi, B., "Completeness Theorems for the Abadi-Rogaway Logic of Encrypted Expressions," *Journal of Computer Security*, vol. 12, no. 1, pp. 99–129, 2004.

[74] Naor, D., Naor, M., and Lotspiech, J., "Revocation and Tracing Schemes for Stateless Receivers," in *CRYPTO*, pp. 41–62, 2001.

[75] Naor, M., "Bit Commitment Using Pseudorandomness," *J. Cryptology*, vol. 4, no. 2, pp. 151–158, 1991.

[76] Naor, M. and Nissim, K., "Certificate Revocation and Certificate Update," in *USENIX Security Symposium*, 1998.

[77] Nisan, N., "Pseudorandom generators for space-bounded computation," *Combinatorica*, vol. 12, no. 4, pp. 449–461, 1992.

[78] NIST, "SHA-3: Cryptographic Hash Algorithm Competition." National Institute of Standards and Technology, 2008. Available at `http://csrc.nist.gov/groups/ST/hash/sha-3/index.html`.

[79] Paterson, K. G. and Yau, A. K. L., "Cryptography in Theory and Practice: The Case of Encryption in IPsec," in *EUROCRYPT*, pp. 12–29, 2006.

[80] Pirretti, M., Traynor, P., McDaniel, P., and Waters, B., "Secure attribute-based systems," in *ACM Conference on Computer and Communications Security*, pp. 99–112, 2006.

[81] Raeburn, K., "Advanced Encryption Standard (AES) Encryption for Kerberos 5," *Network Working Group. Request for Comments: 3962*, 2005. Available at `http://www.ietf.org/rfc/rfc3962.txt`.

[82] Raeburn, K., "Encryption and Checksum Specifications for Kerberos 5," *Network Working Group. Request for Comments: 3961*, 2005. Available at `http://www.ietf.org/rfc/rfc3961.txt`.

[83] Sahai, A. and Waters, B., "Fuzzy Identity-Based Encryption," in *EUROCRYPT*, pp. 457–473, 2005.

[84] Shamir, A., "Identity-Based Cryptosystems and Signature Schemes," in *CRYPTO*, pp. 47–53, 1984.

[85] Shen, P. Y., Liu, V., Tang, M., and Caelli, W. J., "An Efficient Public Key Management System: An Application In Vehicular Ad Hoc Networks," in *PACIS*, p. 175, 2011.

[86] Stubblebine, S. G. and Gligor, V. D., "On Message Integrity in Cryptographic Protocols," in *Symposium on Security and Privacy '92*, IEEE, 1992.

[87] Wang, X., Feng, D., Lai, X., and Yu, H., "Collisions for Hash Functions MD4, MD5, HAVAL-128 and RIPEMD." ePrint Archive: Report 2004/199, 2004. Available at `http://eprint.iacr.org/`.

[88] Waters, B., "Efficient Identity-Based Encryption Without Random Oracles," in *EUROCRYPT*, pp. 114–127, 2005.

[89] Wu, T. D., "A Real-World Analysis of Kerberos Password Security," in *NDSS*, 1999.

[90] Yang, P., Kitagawa, T., Hanaoka, G., Zhang, R., Matsuura, K., and Imai, H., "Applying Fujisaki-Okamoto to Identity-Based Encryption," in *AAECC*, pp. 183–192, 2006.

[91] Yao, A. C.-C., "Theory and Applications of Trapdoor Functions (Extended Abstract)," in *FOCS*, pp. 80–91, 1982.

[92] Yu, T., "The Kerberos Network Authentication Service (version 5)." IETF Internet draft. Request for Comments: 1510, 2006. Available at `http://www.ietf.org/rfc/rfc1510.txt`.

[93] Yu, T., Hartman, S., and Raeburn, K., "The Perils of Unauthenticated Encryption: Kerberos Version 4.," in *NDSS '04*, The Internet Society, 2004.

[94] Zhang, X., Chang, K., Xiong, H., Wen, Y., Shi, G., and Wang, G., "Towards name-based trust and security for content-centric network," in *ICNP*, pp. 1–6, 2011.

VITA

Virendra Kumar is a Ph.D. student in the School of Computer Science, College of Computing at Georgia Institute of Technology. His primary research interests are in Cryptography and Information Security. Before coming to Georgia Tech, he completed his B.Tech. from the Department of Electrical Engineering at Indian Institute of Technology (BHU), Varanasi, India. He was born in a small town in the state of Bihar in India.